EXHIBIT I

UNITED STATES DISTRICT COURT
FOR THE EASTERN DISTRICT OF TEXAS
MARSHALL DIVISION

HEADWATER RESEARCH LLC, *Plaintiff*, v. SAMSUNG ELECTRONICS CO., LTD., SAMSUNG ELECTRONICS AMERICA, INC., *Defendants*.	Civil Action No. 2:22-cv-00422-JRG-RSP **JURY DEMANDED**

SAMSUNG ELECTRONICS CO. LTD. AND SAMSUNG ELECTRONICS AMERICA, INC.'S P.R. 3-3 AND 3-4 INVALIDITY AND PATENT INELIGIBILITY CONTENTIONS

Defendants Samsung Electronics Co., Ltd. and Samsung Electronics America, Inc. (collectively, "Defendants" or "Samsung") hereby provide the following Preliminary Invalidity Contentions ("Contentions") to Plaintiff Headwater Research LLC ("Plaintiff" or "Headwater") for U.S. Patent Nos. 9,137,701 ("the '701 patent"), 9,271,184 ("the '184 patent"), 9,521,578 ("the '578 patent"), 9,277,445 ("the '445 patent"), 11,495,224 ("the '224 patent"), 9,143,976 ("the '976 patent"), 9,277,433 ("the '433 patent"), 9,609,544 ("the '544 patent"), and 10,237,773 ("the '773 patent") (collectively, the "Asserted Patents").

I. PRELIMINARY STATEMENT AND RESERVATION OF RIGHTS

In its Infringement Contentions dated February 28, 2023, Headwater asserted the following 198 claims (the "Asserted Claims"):

- Claims 1-25 of the '701 patent;
- Claims 1-20 of the '184 patent;
- Claims 1-22 of the '578 patent;
- Claims 1-20, 22-26 of the '445 patent;

- Claims 1-17 of the ’224 patent;
- Claims 1-29 of the ’976 patent;
- Claims 1-20 of the ’433 patent;
- Claims 1-8 and 11-23 of the ’544 patent; and
- Claims 1-14 and 16-20 of the ’773 patent.

Samsung does not provide any Contentions directed to claims that Headwater has not asserted for purposes of infringement. To the extent Headwater may be permitted to assert additional claims in the future, Samsung reserves all rights to disclose new or supplemental contentions regarding such claims.

Because the same claim scope must apply for both infringement and invalidity, these Contentions are based on Headwater’s assertions in its Infringement Contentions. Samsung does not thereby implicitly or explicitly agree with Headwater’s construction of the claims. Samsung reserves all rights to disclose new or supplemental invalidity contentions, including to address any construction of the claims rendered by the Court, changed theories of infringement, and any evidence obtained during the course of discovery.

Subject to the rights reserved in these Contentions, all Asserted Claims are invalid under at least one or more of 35 U.S.C. §§ 101, 102, 103, and/or 112. The Asserted Claims are invalid because they are anticipated and/or rendered obvious under 35 U.S.C. §§ 102 and 103. If Headwater contends or a fact-finder finds that one or more limitations of the Asserted Claims are not disclosed in the prior art identified as anticipatory, Samsung reserves the right to assert obviousness based on the identified references and/or to identify other references that would have rendered obvious the allegedly missing limitation. Furthermore, the obviousness combinations of references provided below and in the accompanying claim charts under 35 U.S.C. § 103 are exemplary only and are not intended to be exhaustive. If or when Headwater challenges the disclosure of any of these references with respect to particular limitations of the Asserted Claims, Samsung reserves the right to supplement these Contentions to assert additional or different bases for obviousness. Samsung reserves the right to use any combination of the references set forth in these Contentions to demonstrate the obviousness of the Asserted Claims. Additionally, certain claims of the Asserted Patents are invalid for failure to comply with the written description, enablement, and definiteness requirements of 35 U.S.C. § 112. The Asserted Claims are also invalid for lack of patentable subject matter under 35 U.S.C. § 101.

Samsung expressly reserves the right to amend, correct, and/or supplement these Contentions in accordance with the Procedural Schedule governing these cases.

* * *

These Contentions reflect Samsung's knowledge, investigation, and discovery as of the date of service. Samsung reserves the right to supplement these Contentions as appropriate and for any permissible reason. For example, pursuant to the Procedural Schedule, Samsung reserves the right to supplement these Contentions after subsequent case events, including any disclosure by Headwater of amended or supplemental infringement contentions, any ruling by the Court on claim construction, or in response to arguments made and positions taken by Headwater during fact and expert discovery. Samsung also reserves the right to supplement these Contentions if it becomes aware of additional prior art, becomes aware of additional features of the prior art references cited below, or becomes aware of any other relevant information through discovery, including non-party discovery, or otherwise. Samsung also reserves the right to modify or supplement its Contentions based on the Court's construction of the claims.

In addition to the charts attached hereto, Samsung expressly incorporates by reference, as if expressly set forth in these Contentions, all invalidity positions, prior art, and claim charts asserted against Headwater in any Headwater lawsuit or IPR proceeding by Samsung, prior defendants, petitioners, and potential or actual licensees to the Asserted Patents. Samsung also incorporates any future discovery responses and expert reports in such litigations or proceedings.

Samsung's citations to disclosures in any particular prior art reference are not (and are not intended to be) exhaustive but rather illustrative. Samsung reserves the right to rely on uncited portions of the prior art references and on other publications and expert testimony as aids in understanding and interpreting the cited portions, as providing context thereto, as additional evidence that the prior art discloses a claim limitation or the alleged invention as a whole, as evidence of the state of the art at a particular time, as evidence of the obviousness factor of contemporaneous development by others, and as evidence of motivation to combine. Samsung also reserves the right to rely on uncited portions of the prior art references, other publications, and testimony, including expert testimony, to establish bases for combination of prior art references that render the charted claims obvious. Due to the related nature of the Asserted Patents, Samsung also reserves the right to rely on any cited portions of a prior art reference for one Asserted Patent against all Asserted Patents. Samsung also reserves the right to rely upon any documentary or testimonial evidence of the existence of any systems that embodied or practiced the disclosures found in the accompanying invalidity charts, for example as discussed in the prior art references cited herein, as such systems may qualify as prior art under 35 U.S.C. § 102(g).[1]

Samsung intends to rely on admissions concerning the scope of the prior art relevant to the Asserted Patents found in, *inter alia*: the patent prosecution histories for the Asserted Patents and related patents and/or patent applications (including all prior art cited therein); any deposition testimony of the named inventors on the Asserted Patents and related patents and/or patent

[1] Citations herein refer to the pre-AIA version of Title 35 of the U.S. Code.

applications in this matter or any other matter; evidence and testimony relating to the level of skill in the art; and the papers filed and any evidence submitted by Headwater in connection with this matter.

Samsung reserves the right to assert that the Asserted Claims are invalid under 35 U.S.C. § 102(f) in the event Samsung obtains additional evidence that the inventors named in any of the Asserted Patents did not invent the subject matter claimed therein. Should Samsung obtain such evidence, it will provide the name of the person(s) from whom and the circumstances under which the alleged invention or any part of it was derived.

These Contentions are not intended to include or otherwise reflect Samsung's claim interpretations. Because the Court has not yet construed any of the claims in this litigation, Samsung bases these Contentions at least on its present understanding of Headwater's view and application of the claim scope, to the extent that view can be inferred from Headwater's actual and/or apparent application of those claims. But Samsung does not adopt any constructions or interpretations impliedly or expressly in these Contentions. Moreover, Samsung's Contentions may reflect alternative positions as to claim construction and scope.

For the purposes of these Contentions, Samsung has made assumptions regarding possible meanings of indefinite claim terms. By making these assumptions, Samsung does not admit that any claim language satisfies 35 U.S.C. § 112. Similarly, the use of asserted claim terms herein should not be understood to mean that such terms, as used in the Asserted Patents or claims thereof, are definite or otherwise comply with the conditions of patentability under 35 U.S.C. § 112. Likewise, the use of asserted claim terms herein should not be understood to suggest or imply a common, usual, ordinary, customary, plain, or accepted meaning in the art for any such terms.

By providing these Contentions, Samsung is not waiving nor limiting its rights to make arguments in the future about the proper scope of the claims or to advance alternative constructions to those Headwater advocates. Samsung expressly reserves the right to argue for such alternative claim constructions during this litigation and to supplement these Contentions after the Court has issued a claim construction ruling.

Samsung's factual investigations, including its investigation of prior art and grounds for invalidity, is ongoing. Further, Samsung's invalidity positions will be the subject of expert testimony. Samsung reserves the right to supplement these Contentions, including, without limitation, adding additional prior art and grounds of invalidity in accordance with the Federal Rules of Civil Procedure and Procedural Schedule in these cases, or otherwise.

II. PERSON HAVING ORDINARY SKILL IN THE ART

A person of ordinary skill in the art ("POSITA" or "POSA"), in or about 2009, would have had at least a Bachelors's degree in Electrical Engineering, Computer Science, or equivalent, and 3-5 years of experience with networking, power consumption of networked computing devices, and/or wireless digital communications systems. *See, e.g.*, '976 pat. at 12:47-13:9, 45:5-9, 93:31-42. Additional education might compensate for less experience, and vice versa.

III. IDENTIFICATION OF RELEVANT PRIOR ART

A. Priority Dates

Headwater's Infringement Contentions seemingly allege that all nine Asserted Patents can claim priority to U.S. Provisional Application Ser. No. 61/206,354, filed on Jan. 28, 2009, entitled "Services Policy Communication System and Method" ("the '354 application").[2] Headwater has

[2] While Headwater's Infringement Contentions do not specifically identify the provisional application to which the Asserted Patents allegedly claim priority, Headwater's Infringement Contentions assert that all nine Asserted Patents are entitlted to a January 28, 2009 priority date—

not met its burden of establishing that the claims of the Asserted Patents are entitled to this priority date. *See In re Magnum Oil Tools Int'l, Ltd.*, 829 F.3d 1364, 1376 (Fed. Cir. 2016) (A "patentee bears the burden of establishing that its claimed invention is entitled to an earlier priority date.").

For a patent to claim priority back to the filing date of a particular patent application, all the limitations must appear in the specification of the application to which priority is claimed. However, the '354 application's specification clearly fails to disclose at least the following asserted independent claim limitations, including but not limited to those limitations reciting a "foreground," "background," a "device state," a "power control state," or "Internet activity access controls to aggregate network activitiy," and thus fails to disclose any claims depending therefrom:

1. '701 patent

- **Claim 1:** "determine, for a first end-user application capable of running in a background state and capable of running as a foreground application, whether the application is running in a background state or as a foreground application, and control, via an application program interface (API), application access for Internet service activities provided through the WWAN modem and the WLAN modem, to, based on a first differential traffic control policy, selectively block and allow access by the first end-user application to the WWAN modem at a time when data for Internet service activities is communicated through a WWAN modem connection to the at least one WWAN";
- **Claim 1:** "wherein the access is selectively blocked based on a determination that the first end-user application is running in a background state, and wherein the access is selectively allowed based on a determination that the first end-user application is running as a foreground application."

2. '184 patent

- **Claim 1:** "classify whether a particular application associated with an Internet service access request, and capable of both interacting with a user in a user interface foreground of the device, and at least some Internet service activities when not interacting with a user in the device user interface foreground";

the date on which the '354 application was filed. *See* Headwater's P.R. 3-1(e) disclosures. To the extent that Headwater later asserts it is relying on a different application for priority, Samsung expressly reserves the right to modify these Contentions accordingly.

- **Claim 1:** "is interacting with the user in the device user interface foreground"; and

- **Claim 1:** "apply a differential traffic control policy to the Internet service access request, based on (i) whether the application is classified as interacting with the user, and (ii) a differential traffic control policy list distinguishing between a first one or more applications resident on the device and a second one or more applications resident on the device, such that, the one or more processors are operable to, in a first state wherein the particular application is one of the first one or more applications, and the particular application is not classified as interacting with a user in the device user interface foreground, block the Internet service access request."

3. '578 patent

- Claim 1: "a user interface to allow a user to set one or more of a plurality of aspects of the differential traffic control policy to select one or more applications that are only allowed to utilize the at least one WWAN for Internet service activities when those applications are classified as interacting with a user in the device user interface foreground";

- Claim 1: "one or more processors configured to implement an application program interface (API) that allows a particular application to access one or more aspects of the differential traffic control policy applicable to that application, including whether the user-settable aspects of the policy only allow the particular application to utilize the at least one WWAN for Internet service activities when the particular application is classified as interacting with a user in the device user interface foreground."

4. '445 patent

- **Claim 1:** "classify, as a second classification, whether a particular application associated with an Internet service access request, and capable of both interacting with a user in a user interface foreground of the device";

- **Claim 1:** "at least some Internet service activities when not interacting with a user in the device user interface foreground, is interacting with the user in the device user interface foreground"; and

- **Claim 1:** "block the Internet service access request in a first state of the first and second classifications, wherein data for Internet service activities is classified as to be provided through the WWAN modem, and the particular application is not classified as interacting with a user in the device user interface foreground."

5. '224 patent

- **Claim 1:** "classify, as a first classification for each given one of the network service usage activities, a classification based on the application associated with the given network service usage activity and that allows for a differential network access control, wherein the differential network access control comprises a set of service usage control policies applicable when a network service is available via the at least one wireless modem, including at least a first policy that allows the given network service usage activity to currently communicate data with a network destination via the at least one wireless modem, and a second policy that defers data communication associated with the given network service usage activity until a device state change occurs";

- **Claim 1:** "associate each given one of the network service usage activities with a service usage control policy dynamically selected from the set of service usage control policies, based on the first classification of the given network service usage activity and at least one device state."

6. '976 patent

- **Claim 1:** "one or more processors configured to classify, for a first end-user application capable of interacting in the device display foreground with a user and capable of at least some Internet service activity when not interacting in the device display foreground with the user, whether or not the first end-user application, when running, is interacting in the device display foreground with the user";

- **Claim 1:** "for a time period when data for Internet service activities is communicated through a WWAN modem connection to the at least one WWAN, apply a first differential traffic control policy to Internet service activity on behalf of the first end-user application, such that Internet service activity on behalf of the first end-user application is disallowed when the one or more processors classify the first end-user application as not interacting in the device display foreground with the user";

- **Claim 1:** "a first network access condition that indicates the unavailability to the first end-user application, when the first end-user application is classified as not interacting in the device display foreground with the user, of Internet data service that is available via the WWAN modem"; and

- **Claim 1:** "a second network access condition that indicates the availability to the first end-user application, when the first end-user application is classified as interacting in the device display foreground with the user, of Internet data service that is available via the WWAN modem."

7. '433 patent

- **Claim 1:** "when the one or more Internet activity access controls are to be applied, apply the one or more Internet activity access controls to aggregate network activity for the first Internet access request with network activity for one or more other data communication requests, which are otherwise not associated with the first end-user application, before allowing network activity in association with the first Internet access request."

8. '544 patent

- **Claim 1:** "a processor that executes instructions to associate network service usage activity, on behalf of a first device application, and that occurs when the first device application is not in the foreground of user interaction, with a network service usage control policy";

- **Claim 1:** "set an application state indicating whether the first device application, associated with a particular network service usage activity, is in the foreground of user interaction"; and

- **Claim 1:** "dynamically determine whether to apply the network service usage control policy to the particular network service usage activity, based on the application state and based on a power control state."

9. '773 patent

- **Claim 1:** "receive respective requests from a plurality of applications on the device to access the WWAN for background network service usage activities associated with the respective applications";

- **Claim 1:** "based at least in part on a current WWAN network busy state, select a corresponding current service usage control policy for the background network service usage activities";

- **Claim 1:** "determine respective deferred time slots for each of the background network service usage activities based on the current service usage control policy, wherein at least one such service usage control policy specifies that during a time when the current WWAN network busy state indicates network congestion, a selected subset of the background network service usage activities are deferred until network congestion is no longer indicated"; and

- **Claim 1:** "allow each of the background network service usage activities to access the WWAN during the respective deferred time slot for that background network service usage activity."

Thus, the Asserted Claims' cannot claim priority to January 28, 2009. Nevertheless, even under Headwater's incorrect priority date, the Asserted Patents are invalid.

To the extent it is later argued by Headwater, or otherwise determined that a different priority date applies, Samsung reserves the right to amend these Contentions accordingly.

B. Prior Art Patent Publications

Based on their investigation to date, Samsung has provided in the list below the prior art patent publications presently known to Samsung that it contends anticipate and/or render obvious the Asserted Claims. The prior art identified in these Contentions discloses (i.e., anticipates and/or renders obvious) the elements of the Asserted Claims either explicitly or inherently. Similarly, the prior art patent publications listed on the face of the Asserted Patents discloses (i.e., anticipates and/or renders obvious) the elements of the Asserted Claims either explicitly or inherently, and Samsung reserves the right to rely on any such reference.

Prior-art patents or publications included in these Contentions may be related (such as a divisional, continuation, continuation-in-part, parent, or child) to earlier or later-filed patents or publications, may have counterparts filed in other jurisdictions, or may incorporate (or be incorporated by) other patents or publications by reference. The listed patents or publications are intended to be representative of these other patents or publications to the extent they exist. Samsung accordingly reserves the right to modify, amend, or supplement these Contentions with these related patents or publications, as well as other prior art references, upon further investigation. Additionally, any reference in these Contentions, including the appendices and exhibits thereto, to a specific subsection or subsections of 35 U.S.C. § 102, is merely exemplary, and Samsung expressly reserves the right to rely on additional or other sections of 35 U.S.C. § 102, as appropriate. If Headwater asserts that one or more of these references or systems fails to disclose one or more elements of a claim, Samsung reserves the right to also use those references to invalidate the claim under 35 U.S.C. § 103.

Discovery is ongoing, and Samsung's prior art investigation and third-party discovery is therefore not yet complete. Samsung reserves the right to present additional items of prior art under 35 U.S.C. §§ 102 and/or 103 that are located during the course of discovery or further investigation. For example, Samsung expects to receive documents from additional third parties either through informal requests or under subpoenas that are believed to have knowledge, documentation, and/or corroborating evidence concerning some of the prior art listed and discussed below. These third parties include without limitation the authors, inventors, or assignees of the references listed in these disclosures.

Patent Publication	Publication/Issue Date
U.S. Patent No. 7,260,635 B2	Aug. 21, 2007
U.S Patent Application Pub. No. 2002/0166117 A1	Nov. 7, 2002
U.S Patent Application Pub. No. 2006/0039354 A1	Feb. 23, 2006
U.S Patent Application Pub. No. 2006/0059556 A1	Mar. 16, 2006
U.S Patent Application Pub. No. 2010/0017506 A1	Jan. 21, 2010
U.S Patent Application Pub. No. 2011/0019557 A1	Jan. 27, 2011
U.S. Patent No. 7,069,330 B1	Jun. 27, 2006
U.S. Patent No. 8,397,087 B1	Mar. 12, 2013
U.S. Patent No. 8,667,513 B1	Mar. 4, 2014
U.S Patent Application Pub. No. 2005/0026654 A1	Feb. 3, 2005
U.S. Patent No. 5,987,611	Nov. 16, 1999
U.S Patent Application Pub. No. 2012/0102504 A1	Apr. 26, 2012
U.S. Patent No. 9,098,333 B1	Aug. 4, 2015
U.S Patent Application Pub. No. 2012/0265897 A	Oct. 18, 2012
U.S Patent Application Pub. No. 2009/0190471 A1	Jul. 30, 2009
U.S. Patent No. 8,320,272 B2	Nov. 27, 2012

U.S. Patent No. 8,437,748 B2	May 7, 2013
U.S Patent Application Pub. No. 2008/0085717 A1	Apr. 10, 2008
U.S Patent Application Pub. No. 2010/0112955 A1	May 6, 2010
European Patent Application EP 1 484 871 A1	Dec. 8 2004
KR 2009/0036186 A	Apr. 14, 2009
U.S. Patent No. 9,398,103 B2	Jul. 19, 2016
U.S. Patent Application Pub. No. 2002/0133598 A1	Sep. 19, 2002
U.S. Patent Application Pub. No. 2008/0282080 A1	Nov. 13, 2008
International Publication No WO 2008/043278 A1	Apr. 17, 2008
U.S. Patent No. 9,008,673 B1	Apr. 14, 2015
U.S. Patent Application Pub. No. 2009/0028127 A1	Jan. 29, 2009
U.S. Patent No. 6,898,654 B1	May 24, 2005
U.S. Patent Application Pub. No. 2006/0053113 A1	Mar. 9, 2006
International Publication No. WO 2005/015379 A1	Feb. 2, 2008
U.S. Patent No. 8,099,078 B2	Jan. 17, 2012
U.S. Patent Application Pub. No. 2006/0229090 A1	Oct. 12, 2006
Australian Application Pub. No. 2003255114 B2	Aug. 13, 2003

U.S. Patent Application Pub. No. 2007/0038763 A1	Feb. 15, 2007
U.S. Patent Application Pub. No. 2009/0207817 A1	Aug. 20, 2009
U.S. Patent No. 6,578,077 B1	Jun. 10, 2003
U.S. Patent Application Pub. No. 2008/0080457 A1	Apr. 3, 2008
U.S. Patent Application Pub. No. 2008/0084977 A1	Apr. 10, 2008
U.S. Patent Application Pub. No. 2005/0108075 A1	May 19, 2005
U.S. Patent No. 7,290,034 B2	Oct. 30, 2007
U.S. Patent No. 8,028,060 B1	Sep. 27, 2011
U.S. Patent No. 8,260,253 B2	Sep. 4, 2012
U.S. Patent No. 9,516,129 B2	Dec. 6, 2016
U.S. Patent Application Pub. No. 2007/0162582 A1	Jul. 12, 2007
U.S. Patent Application Pub. No. 2008/0080458 A1	Apr. 3, 2008
U.S. Patent Application Pub. No. 2009/0217065 A1	Aug. 27, 2009
U.S. Patent Application Pub. No. 2009/0307696 A1	Dec. 10, 2009
U.S. Patent Application Pub. No. 2010/0077035 A1	Mar. 25, 2010
U.S. Patent Application Pub. No. 2010/0115048 A1	May 6, 2010
U.S. Patent Application Pub. No. 2011/0185202 A1	Jul. 28, 2011

U.S. Patent Application Pub. No. 2012/0023236 A1	Jan. 26, 2012
U.S. Patent Application Pub. No. 2012/0185577 A1	Jul. 19, 2012
U.S. Patent Application Pub. No. 2012/0260118 A1	Oct. 11, 2012
U.S. Patent Application Pub. No. 2012/0284620 A1	Nov. 8, 2012
U.S. Patent Application Pub. No. 2011/0249668 A1	Oct. 13, 2011
U.S. Patent Application Pub. No. 2011/0312283 A1	Dec. 22, 2011
U.S. Patent Application Pub. No. 2012/0157038 A1	Jun. 21, 2012
U.S. Patent No. 7,821,985 B2	Oct. 26, 2010
U.S. Patent Application Pub. No. 2009/0119773 A1	May 7, 2009
U.S. Patent No. 7,751,330 B2	Jul. 6, 2010
U.S. Patent Application Pub. No. 2011/0019574 A1	Jan. 27, 2011
U.S. Patent No. 8,135,443 B2	Mar. 13, 2012
U.S. Patent Application Pub. No. 2011/0314151 A1	Dec. 22, 2011
U.S. Patent No. 8,626,115 B2	Jan. 7, 2014
U.S. Patent No. 8,589,541 B2	Nov. 19, 2013
U.S. Patent No. 8,429,409 B1	Apr. 23, 2013
U.S. Patent No. 6,185,576 B2	Feb. 6, 2001
U.S. Patent No. 8,666,364 B2	Mar. 4, 2014

U.S. Patent No. 8,634,821 B2	Jan. 21, 2014
U.S. Patent Application Pub. No. 2013/0117382 A1	May 9, 2013
U.S. Patent Application Pub. No. 2013/0103376 A1	Apr. 25, 2013
U.S. Patent Application Pub. No. 2013/0149994 A1	Jun. 13, 2013
U.S. Patent Application Pub. No. 2013/0111572 A1	May 2, 2013
U.S. Patent Application Pub. No. 2012/0101952 A1	Apr. 26, 2012
U.S. Patent Application Pub. No. 2013/0183937 A1	Jul. 18, 2013
U.S. Patent Application Pub. No. 2015/0207760 A1	Jul. 23, 2015
International Publication No. WO 2012/047275A1	Apr. 12, 2012
JP 2010/261635 A	Nov. 18, 2010

C. Prior Art Non-Patent Publications[3,4]

Non-Patent Publication	Publication Date
Wireless Wakeups Revisited: Energy Management for VoIP over Wi-Fi Smartphones	June 2007
How Can I Run an Application at a Higher Priority?	Mar. 19, 2000
Wake on Wireless: An Event Driven Energy Saving Strategy for Battery Operated Devices	Sep. 2002
Power Management in Mobile and Pervasive Computing Systems	November 2005
BlackBerry Pearl 9105 Smartphone	2010
Nokia N95 8GB User Guide	2007
Power Management Techniques for Mobile Communication	1998
Internet Traffic Manager	Aug. 11, 2009
Nexus One User's Guide	Mar. 15, 2010
iPhone User Guide For iPhone OS 3.1 Software	2009
Nokia E90 Communicator User Guide	2007
Nokia N97 User Guide	2009

[3] Any discussion of a non-patent publication in either Section III.C or in one of the claim charts included herewith that discloses a corresponding product or system shall also apply with equal force to the underlying product or system. In other words, both the non-patent publication and the underlying product or system themselves qualify as prior art in the context that they are used herein.

[4] Discovery is currently ongoing, and Samsung will supplement these Contentions with respect to the public availability, as necessary, of any non-patent publication if and when more information becomes available. Indeed, Samsung expects to receive documents from third parties either through informal requests or under subpoenas that are believed to have knowledge, documentation, and/or corroborating evidence concerning the public availability of the identified non-patent publications.

Developing Software for Symbian OS	2006

Moreover, the prior art non-patent publications listed on the face of the Asserted Patents discloses (i.e., anticipates and/or renders obvious) the elements of the Asserted Claims either explicitly or inherently, and Samsung reserves the right to rely on any such reference.

D. Prior Art Systems and/or Knowledge

The Asserted Claims are invalid under 35 U.S.C. §§ 102 and/or 103 based on prior art items offered for sale or publicly used or known or prior inventions, such as prior art products, including systems embodying any alleged inventions or structures described in, and/or any knowledge disclosed by or referred to in, any of the prior art patents or prior art publications identified above in Sections III.B and III.C. Because Samsung has not yet completed discovery in this case, Samsung reserves the right to supplement theses Contentions with facts, documents, or other information learned at a later point through third-party discovery or further investigation. For example, Samsung expects to receive documents from additional third parties either through informal requests or under subpoenas that are believed to have knowledge, documentation, and/or corroborating evidence concerning some of the prior art listed above and below and/or additional prior art. These third parties include without limitation the authors, inventors, or assignees of the references listed in these Contentions. In addition, Samsung reserves the right to assert invalidity under other sections of 35 U.S.C. § 102 to the extent that discovery or further investigation yield information forming the basis for such invalidity.

Moreover, all of the systems and products listed below qualify as prior art to each of the Asserted Patents under at least pre-AIA 35 U.S.C. §§ 102(a)/(b). Such systems and products were known, used, offered for sale, and/or sold in the United States prior to the appropriate priority date corresponding to each of the Asserted Patents.

Devices
Android Devices (~~including~~e.g., HTC Dream/T-Mobile G1, Samsung GT-I7500 Galaxy, Nexus One, and emulators)[5]
Apple Devices (including iPhone, iPhone 3G, iPhone 3GS, and emulators)
Symbian Devices (including Nokia E90, N95, N97, E72, and emulators)
Windows Mobile Devices (including devices with Windows Mobile installed and emulators)
Windows XP Devices (including devices with Windows XP installed and emulators)

Operating Systems
Android including Android 1.0 (released September 2008), Android 1.1 (released February 2009), Android Cupcake (1.5) (released April 2009), Android Donut (1.6) (released September 2009), Android Eclair (2.0, 2.0.1, 2.1) (released October 2009 - January 2010), Android Froyo (2.2) (released May 20, 2010)[6]
iPhone OS including iPhone OS 1.0 (released June 29, 2007), iPhone OS 2.0 (released July 11, 2008), iPhone OS 3.0 (released June 19, 2009)
Symbian OS including Symbian OS 9.1 (released 2005), Symbian OS 9.2, 9.3 (released 2006), Symbian OS 9.4 (released 2007), and associated platforms including S60 3rd Edition (released 2001/2002), S60 3rd Edition, Feature Pack 1 and Feature Pack 2, and S60 5th Edition (released October 2008)
Windows Mobile including Windows Mobile 6.1 (released April 1, 2008) and Windows Mobile 6.5 (released May 11, 2009)
Windows XP including its various editions such as Home Edition and Tablet PC Edition (Windows XP was initially released October 2001)

Applications
JuiceDefender (including JuiceDefender and associated add-on application, UltimateJuice)
GreenPower

[5] *See, e.g.*, SAMSUNG_PRIORART0000001-334; SAMSUNG_PRIORART0005174-76; SAMSUNG_PRIORART0005177-317; SAMSUNG_PRIORART0005416-19; SAMSUNG_PRIORART0005420-23; SAMSUNG_PRIORART0005424-28; SAMSUNG_PRIORART0005429-44; SAMSUNG_PRIORART0005445-48; SAMSUNG_PRIORART0005449-52; SAMSUNG_PRIORART0005453-57; SAMSUNG_PRIORART0005458-71; SAMSUNG_PRIORART0005472-77; SAMSUNG_PRIORART0005478-84; SAMSUNG_PRIORART0005485-86; SAMSUNG_PRIORART0005488-5624; Section IV.B.2, *infra.*

[6] *E.g.*, GOOG-HEADWATER-000000001-123; SAMSUNG_PRIORART0005042-5487; Section IV.B.2, *infra*. For Android versions 1.0-2.2, code from earlier versions carried over into later versions. Thus, any code cited from an earlier version of Android also exists in the later version.

Microsoft Applications (including Microsoft Outlook Mobile, Windows Live, and MyPhone)
Apple Applications (including Mail, Calendar, MobileMe, Backup, and Newsstand API)
Citrix Applications (including XenApp, XenDesktop, NetScaler, Citrix Receiver, Citrix Cloud Bridge)

The Federal Circuit has held that "[t]he proper test for the public use prong of the [pre-AIA] § 102(b) statutory bar is whether the purported use: (1) was accessible to the public; or (2) was commercially exploited." *See Invitrogen Corp. v. Biocrest Mfg. L.P.,* 424 F.3d 1374, 1380 (Fed. Cir. 2005). Additionally, the on-sale bar of § 102(b) is triggered when the invention is both (1) the subject of a commercial offer for sale not primarily for experimental purposes and (2) ready for patenting. *Pfaff v. Wells Elecs., Inc*., 525 U.S. 55, 67 (1998). Each of the systems and products listed above meets these criteria.

The above discussion is not exclusive. Samsung reserves the right to rely on both the listed products as well as other products that may become known and/or relevant during the course of this matter. Samsung also reserves the right to rely on forthcoming testimony and/or declarations from third-party witnesses, including but not limited to witnesses from Google, Apple, Microsoft, Nokia, HMD, and Citrix relating to the above products, applications, and operating systems.

Any citation to one or more of these prior art references, or other prior art references regarding any method or system, should be construed to constitute not only a citation to the prior art reference itself but also a reference to the system itself. Discovery is ongoing in this case, and Samsung will supplement these Contentions if and when more information becomes available. For example, Defendants are already in the process of taking discovery from non-parties including Nokia, HMD, Citrix, Google, Apple, and Microsoft. Accordingly, Defendants reserve the right to modify, amend, and/or supplement these contentions as information becomes available from non-parties.

E. Prior Art Under 35 U.S.C. §§ 102(f) and 102(g)

Each prior art patent, publication, or product identified above was either effectively filed or issued (for patents), published (for publications) or known, used, offered for sale or sold (for products) before either the earliest claimed priority date of the Asserted Patents to which it is applied for invalidity, and none appears to have been abandoned, suppressed, or concealed, so each such reference also constitutes evidence of prior invention pursuant to 35 U.S.C. § 102(g), if it is in the U.S. The persons or entities involved with each such invention include the named inventors on the above-identified patents, the authors listed on the above-identified publications, and the entities and individuals identified in connection with the above-identified products.

Because Samsung has not yet completed discovery in this case, including taking depositions of the named inventors of the Asserted Patents, reviewing Headwater's productions, and seeking discovery of prior inventions by third parties, Samsung reserves the right to supplement these Contentions with facts, documents, or other information learned at a later point through discovery or further investigation.

IV. ANTICIPATION AND OBVIOUSNESS (35 U.S.C. §§ 102 AND 103)

The Asserted Claims are anticipated by and/or rendered obvious in view of one or more items of prior art identified in these Contentions, alone and/or in combination. Based on its investigation to date, Samsung has provided in the lists above the prior art presently known to Samsung that anticipates and/or renders obvious the Asserted Claims under at least Headwater's actual and/or apparent application of those claims. The prior art identified in these Contentions discloses (i.e., anticipates and/or renders obvious) the elements of the Asserted Claims either explicitly or inherently.

Prior art patents or publications included in these Contentions may be related (such as a divisional, continuation, continuation-in-part, parent, or child) to earlier or later-filed patents or publications, may have counterparts filed in other jurisdictions, or may incorporate (or be incorporated by) other patents or publications by reference. The listed patents or publications are intended to be representative of these other patents or publications to the extent they exist. Samsung accordingly reserves the right to modify, amend, or supplement these Contentions with these related patents or publications, as well as other prior art references, upon further investigation. Additionally, any reference in these Contentions, including the appendices and/or exhibits thereto, to a specific subsection or subsections of 35 U.S.C. § 102, is merely exemplary, and Samsung expressly reserves the right to rely on additional or other sections of 35 U.S.C. § 102, as appropriate.

Although Samsung's investigation is ongoing, information available to date indicates that each prior art system disclosed above was at least (1) known or used in this country before the alleged invention of the claimed subject matter of the Asserted Patents; (2) in public use, on sale, or offered for sale in this country more than one year before the effective filing date for the Asserted Patents; or (3) invented and not abandoned, suppressed, or concealed prior to the alleged invention of the Asserted Patents.

Much of the art identified in these Contentions reflects common knowledge and the state of the art prior to the filing or asserted priority dates of the Asserted Patents. As such, the obviousness combinations in these Contentions are intended to be exemplary. There are many possible combinations of the disclosed prior art, and the inclusion of certain exemplary combinations does not exclude other combinations. For example, where a particular contention calls for combining references, any of a number of references can be combined.

Depending on the construction of the claims of the Asserted Patents, and/or positions that Headwater or its expert witnesses may take concerning claim interpretation, infringement, and/or invalidity issues, different ones of the charted prior art references in the Exhibits may be of greater or lesser relevance and different combinations of these references may be implicated. Given the uncertainty, the charts may reflect alternative applications of the prior art against the Asserted Claims.

Citations to particular excerpts from the prior art are likewise exemplary and not exhaustive of the evidentiary support for the invalidity of the Asserted Patents contained in and/or concerning a particular piece of prior art. Samsung may rely on uncited portions of the prior art references, other documents or operational systems, the "Background of the Invention" and other relevant portions of the Asserted Patents, the prosecution histories of the Asserted Patents (including all cited references) and their related patents and applications, and forthcoming fact and expert testimony to provide context to aid in understanding the prior art reference and/or the cited portions of the references. Where Samsung cites to a particular figure in a reference, the citation encompasses the caption and description of the figure and any text relating to or discussing the figure. Likewise, where Samsung cites text referring to a figure, the citation includes the figure as well (and vice versa).

A. Prior Art Under 35 U.S.C. § 102

Samsung contends that at least the primary prior art references (Exs. -01 through -09) and systems (Exs. -10 through -11) identified below, by themselves, anticipate the Asserted Claims:

Exhibits	Primary References and/or Systems
A-01, B-01, C-01, D-01, E-01, F-01, G-01, H-01, I-01	U.S. Patent Publication No. 2008/0080458A1 to Cole ("Cole")
A-02, B-02, C-02, D-02, E-02, F-02, G-02, H-02, I-02	U.S. Patent Publication No. 2007/0038763A1 to Oestvall ("Oestvall")

Exhibits	Primary References and/or Systems
A-03, B-03, C-03, D-03, E-03, F-03, G-03, H-03, I-03	U.S. Patent Publication No. 2006/0039354A1 to Rao et al. ("Rao")
A-04, B-04, C-04, D-04, E-04, F-04, G-04, H-04, I-04	European Patent No. 1,484,871 A1 to Kelz ("Kelz")
A-05, B-05, C-05, D-05, E-05, F-05, G-05, H-05, I-05	U.S. Patent Publication No. 2010/0115048A1 to Scahill ("Scahill")
A-06, B-06, C-06, D-06, E-06, F-06, G-06, H-06, I-06	U.S. Patent No. 8,028,060 B1 to Wyld et al. ("Wyld")
A-07, B-07, C-07, D-07, E-07, F-07, G-07, H-07, I-07	U.S. Patent Publication No. 2009/0217065A1 to Araujo, JR. ("Araujo")
A-08, B-08, C-08, D-08, E-08, F-08, G-08, H-08, I-08	U.S. Patent No. 5,987,611 to Freund ("Freund")
A-09, B-09, C-09, D-09, E-09, F-09, G-09, H-09, I-09	U.S. Patent No. 6,898,654 B1 to Senior et al. ("Senior")
A-10, B-10, C-10, D-10, E-10, F-10, G-10, H-10, I-10	Android Devices
A-11, B-11, C-11, D-11, E-11, F-11, G-11, H-11, I-11	Mobile Devices: Symbian Devices, Apple Devices, Microsoft Mobile Devices, Microsoft XP Devices

Specifically, Samsung contends that at least the references and/or systems in the table above independently anticipate the Asserted Claims under 35 U.S.C. §§ 102(a), (b), (e), (f) and/or (g), as set forth in the charts attached as:

- Exhibits A-01 through Exhibits A-11 for the asserted claims of the '701 patent;

- Exhibits B-01 through Exhibits B-11 for the asserted claims of the ’184 patent;
- Exhibits C-01 through Exhibits C-11 for the asserted claims of the ’578 patent;
- Exhibits D-01 through Exhibits D-11 for the asserted claims of the ’445 patent;
- Exhibits E-01 through Exhibits E-11 for the asserted claims of the ’224 patent;
- Exhibits F-01 through Exhibits F-11 for the asserted claims of the ’976 patent;
- Exhibits G-01 through Exhibits G-11 for the asserted claims of the ’433 patent;
- Exhibits H-01 through Exhibits H-11 for the asserted claims of the ’544 patent; and
- Exhibits I-01 through Exhibits I-11 for the asserted claims of the ’773 patent.

Where an asserted prior art reference in any attached claim charts relies on a claim of priority to assert a critical reference date under pre-AIA 35 U.S.C. § 102 *et seq*. (including pre-AIA § 102(e)), compliance with pre-AIA 35 U.S.C. § 112, first paragraph, or 35 U.S.C. § 112(a), is shown in an appendix to a given claim.

These charts, however, are exemplary. The claimed features are similarly described and suggested in other places (including in all of the documents cited during prosecution of each piece of prior art), and also were present when prior-art systems practicing the described prior art were used before the application that ultimately led to the Asserted Patents. Thus, where patents or other printed materials are disclosed, Samsung reserves the right to also rely on those materials as descriptions of systems, devices, or methods referenced therein, publicly used, and/or on sale or known in the United States. Further, Samsung reserves the right to rely on other evidence of the prior art beyond merely the exemplary references cited in the charts attached as Exhibits.

Where patents or other printed materials are disclosed, Samsung reserves the right to also rely on those materials as descriptions of systems, devices, or methods referenced therein, publicly used, and/or on sale or known in the United States. Samsung reserves the right to also use those references to invalidate the claim under 35 U.S.C. § 103.

B. Prior Art Under 35 U.S.C. § 103

To the extent that a primary reference is deemed, by itself, not to anticipate or render obvious a claim for failing to teach one or more limitations, the claim would nonetheless have been obvious to a POSITA at the time of the invention by the combination of the primary reference with one or more other primary references and/or the knowledge of someone skilled in the art.

Moreover, Exhibits A-A, B-B, C-C, D-D, E-E, F-F, G-G, H-H, and I-I list secondary prior art references and identify, on limitation-by-limitation bases, exemplary disclosures where each secondary reference teaches the limitations of the asserted claims. To the extent that a primary reference is deemed, by itself, not to anticipate or render obvious a claim for failing to teach one or more limitations, the claim would nonetheless have also been obvious to a POSITA at the time of the invention by the additional combination of the primary reference with one or more of the references listed as disclosing those alleged missing limitations in Exhibits A-A, B-B, C-C, D-D, E-E, F-F, G-G, H-H, and I-I.

As such, a POSITA would have been motivated to combine any reference set forth in at least the following charts:

- Exhibit A-01 through Exhibit A-11 and Exhibit A-A for the asserted claims of the ’701 patent;
- Exhibit B-01 through Exhibit B-11 and Exhibit B-B for the asserted claims of the ’184 patent;
- Exhibit C-01 through Exhibit C-11 and Exhibit C-C for the asserted claims of the ’578 patent;
- Exhibit D-01 through Exhibit D-11 and Exhibit D-D for the asserted claims of the ’445 patent;
- Exhibit E-01 through Exhibit E-11 and Exhibit E-E for the asserted claims of the ’224 patent;
- Exhibit F-01 through Exhibit F-11 and Exhibit F-F for the asserted claims of the ’976 patent;

- Exhibit G-01 through Exhibit G-11 and Exhibit G-G for the asserted claims of the ’433 patent;
- Exhibit H-01 through Exhibit H-11 and Exhibit H-H for the asserted claims of the ’544 patent; and
- Exhibit I-01 through Exhibit I-11 and Exhibit I-I for the asserted claims of the ’773 patent.

Such combinations would be achieved, for example, by merely combining the disclosures described in the respective claim charts for each reference.

These charts, however, are exemplary. The claimed features are similarly described and suggested in other places (including in all of the documents cited during prosecution of each piece of prior art), and also were present when prior-art systems practicing the described prior art were used before the application that ultimately led to the Asserted Patents. For example, like the Asserted Patents, U.S. Patent No. 8,666,364 B2 includes the concept of reducing data usage over multiple networks (WLAN, WWAN). Similarly, like the Asserted Patents, U.S. Patent Pub. No. 2012/0101952A1 includes the concept of services level per application. *See* 2012/0101952A1 at [0084] (“Additional embodiments may be provided to manage the level of services delivered to networked devices and to manage such services on a service-by-service basis. For example, various embodiments can be configured to provide cost effective services or service classes that match growing digital networking usage patterns. As a further example, in some embodiments, systems and methods are provided that allow access providers to not only control and bill for basic access, but to control and bill for higher level service delivery on a service-by-service basis. Such service-by-service control and billing can be provided based on service types or on service classes or on a combination thereof. Example services may include rich Internet access and email, application based billing, content distribution, entertainment activities, information or content subscription or gaming.”). Thus, where patents or other printed materials are disclosed, Samsung reserves the right to also rely on those materials as descriptions of systems, devices, or methods referenced therein, publicly used, and/or on sale or known in the United States. Further, Samsung reserves the right to rely on other evidence of the prior art beyond merely the exemplary references cited in the charts attached as Exhibits.

Samsung's assertion that the combinations above render the asserted claims obvious under 35 U.S.C. § 103 is not, and is not intended to be, an admission or suggestion that each reference does not independently anticipate the Asserted Claims under 35 U.S.C. § 102. *See Connell v. Sears, Roebuck & Co.*, 722 F.2d 1542, 1548 (Fed. Cir. 1983) ("'[A]nticipation is the epitome of obviousness.'") (quoting *In re Fracalossi*, 681 F.2d 792, 794 (CCPA 1982)). Further, the fact that certain secondary references are listed solely in Exhibits A-A, B-B, C-C, D-D, E-E, F-F, G-G, H-H, and I-I is not intended to be an admission or suggestion that each individual reference cited therein does not also independently anticipate and/or render obvious the Asserted Claims under 35 U.S.C. §§ 102 and 103. Samsung expressly reserves the right to rely on any secondary reference cited in Exhibits A-A, B-B, C-C, D-D, E-E, F-F, G-G, H-H, and I-I as if it were set forth as a primary reference in Section IV.A, *supra*. Finally, the inclusion of the exemplary combinations in the attached Exhibits and Appendices does not exclude other combinations of prior art disclosed in this or previous sections.

1. Exemplary Combinations

Exemplary combinations of prior art references that render the Asserted Claims invalid as obvious under 35 U.S.C. § 103 are described in:

- Exhibit A-01 through Exhibit A-11 and Exhibit A-A for the asserted claims of the '701 patent;
- Exhibit B-01 through Exhibit B-11 and Exhibit B-B for the asserted claims of the '184 patent;
- Exhibit C-01 through Exhibit C-11 and Exhibit C-C for the asserted claims of the '578 patent;

- Exhibit D-01 through Exhibit D-11 and Exhibit D-D for the asserted claims of the ’445 patent;
- Exhibit E-01 through Exhibit E-11 and Exhibit E-E for the asserted claims of the ’224 patent;
- Exhibit F-01 through Exhibit F-11 and Exhibit F-F for the asserted claims of the ’976 patent;
- Exhibit G-01 through Exhibit G-11 and Exhibit G-G for the asserted claims of the ’433 patent;
- Exhibit H-01 through Exhibit H-11 and Exhibit H-H for the asserted claims of the ’544 patent; and
- Exhibit I-01 through Exhibit I-11 and Exhibit I-I for the asserted claims of the ’773 patent.

Moreover, each prior art reference or system may be combined with (1) information known to persons skilled in the art at the time of the alleged invention; (2) any other anticipatory prior art references or systems; and (3) any of the additional prior art identified above or in the prosecution of the Asserted Patents and related applications.

Below are examples of prior art references and/or systems that would have been combined by one of ordinary skill in the art at the time of the alleged invention. These combinations are merely examples. The Asserted Claims are rendered obvious by:

- Cole alone or in combination with one or more of Oestvall, Montemurro, Rao, Kelz, Scahill, Wyld, Van Megen, Maes, Vals, Araujo, Freund, Malomsoky, D’Amore, Lee, Brisebois, Aleksic, and Senior;
- Oestvall alone or in combination with one or more of Cole, Montemurro, Rao, Kelz, Scahill, Wyld, Van Megen, Maes, Vals, Araujo, Freund, Malomsoky, D’Amore, Lee, Brisebois, Aleksic, and Senior;
- Rao alone or in combination with one or more of Cole, Oestvall, Montemurro, Kelz, Scahill, Wyld, Van Megen, Maes, Vals, Araujo, Freund, Malomsoky, D’Amore, Lee, Brisebois, Aleksic, and Senior;
- Kelz alone or in combination with one or more of Cole, Oestvall, Montemurro, Rao, Scahill, Wyld, Van Megen, Maes, Vals, Araujo, Freund, Malomsoky, D’Amore, Lee, Brisebois, Aleksic, and Senior;

- Scahill alone or in combination with one or more of Cole, Oestvall, Montemurro, Rao, Kelz, Wyld, Van Megen, Maes, Vals, Araujo, Freund, Malomsoky, D'Amore, Lee, Brisebois, Aleksic, and Senior;

- Wyld alone or in combination with one or more of Cole, Oestvall, Montemurro, Rao, Kelz, Scahill, Van Megen, Maes, Vals, Araujo, Freund, Malomsoky, D'Amore, Lee, Brisebois, Aleksic, and Senior;

- Araujo alone or in combination with one or more of Cole, Oestvall, Montemurro, Rao, Kelz, Scahill, Wyld, Van Megen, Maes, Vals, Freund, Malomsoky, D'Amore, Lee, Brisebois, Aleksic, and Senior;

- Freund alone or in combination with one or more of Cole, Oestvall, Montemurro, Rao, Kelz, Scahill, Wyld, Van Megen, Maes, Vals, Araujo, Malomsoky, D'Amore, Lee, Brisebois, Aleksic, and Senior;

- Senior alone or in combination with one or more of Cole, Oestvall, Montemurro, Rao, Kelz, Scahill, Wyld, Van Megen, Maes, Vals, Araujo, Malomsoky, D'Amore, Lee, Brisebois, Aleksic, and Freund;

- Android Devices (including operating systems and applications running therein) with one or more applications alone or in combination with one or more of Cole, Oestvall, Montemurro, Rao, Kelz, Scahill, Wyld, Van Megen, Maes, Vals, Araujo, Freund, Malomsoky, D'Amore, Lee, Brisebois, Aleksic, and Senior;

- Apple Devices (including operating systems and applications running therein) with one or more applications alone or in combination with one or more of Cole, Oestvall, Montemurro, Rao, Kelz, Scahill, Wyld, Van Megen, Maes, Vals, Araujo, Freund, Malomsoky, D'Amore, Lee, Brisebois, Aleksic, and Senior;

- Symbian Devices (including operating systems and applications running therein) alone or in combination with one or more of Cole, Oestvall, Montemurro, Rao, Kelz, Scahill, Wyld, Van Megen, Maes, Vals, Araujo, Freund, Malomsoky, D'Amore, Lee, Brisebois, Aleksic, and Senior;

- Windows Mobile Devices (including operating systems and applications running therein) alone or in combination with one or more of Cole, Oestvall, Montemurro, Rao, Kelz, Scahill, Wyld, Van Megen, Maes, Vals, Araujo, Freund, Malomsoky, D'Amore, Lee, Brisebois, Aleksic, and Senior;

- Windows XP Devices (including operating systems and applications running therein) alone or in combination with one or more of Cole, Oestvall, Montemurro, Rao, Kelz, Scahill, Wyld, Van Megen, Maes, Vals, Araujo, Freund, Malomsoky, D'Amore, Lee, Brisebois, Aleksic, and Senior; and

- Any combination of one or more of Android Devices, Apple Devices, Symbian Devices, Windows Mobile Devices, and Windows XP Devices, including the operating systems and applications running therein.

2. Exemplary Motivations to Combine

To the extent a finder of fact finds that any primary prior art reference does not disclose one or more limitations of an asserted claim, the asserted claim is nevertheless obvious because the allegedly missing limitations contain nothing beyond ordinary improvements. In other words, the asserted claim combines known elements to achieve predictable results or chooses between clear alternatives known to those of skill in the art, particularly in view of the state of the art as reflected in the relevant prior art.

Moreover, as explained above, it would have been obvious to a person of skill in the art at the time of the alleged invention of the asserted claims to combine any primary reference with any combination of other primary references or secondary references so as to practice the asserted claims. To the extent that Headwater argues that any concept claimed in the asserted claims is not disclosed in a primary reference, it would, at a minimum, have been obvious to adapt the primary reference to include the concept or combine it with other primary references or secondary references that disclose the concept. Each concept described and claimed in the Asserted Patents was known to those of skill in the art as available design choices for various network data saving features, battery saving features, and network connectivity management functions.

The Supreme Court has held that "[t]he combination of familiar elements according to known methods is likely to be obvious when it does no more than yield predictable results." *KSR Int'l Co. v. Teleflex Inc.*, 550 U.S. 398, 416 (2007). "When a work is available in one field of endeavor, design incentives and other market forces can prompt variations of it, either in the same field or a different one." *Id.* at 417. As the Supreme Court made clear, "[f]or the same reason, if a technique has been used to improve one device, and a person of ordinary skill in the art would recognize that it would improve similar devices in the same way, using the technique is obvious unless its actual application is beyond his or her skill." *Id.*

To determine whether there is an apparent reason to combine the known elements in the fashion claimed by the patent at issue, a court can "look to interrelated teachings of multiple patents; the effects of demands known to the design community or present in the marketplace; and the background knowledge possessed by a person having ordinary skill in the art." *Id.* at 418. For example, obviousness can be demonstrated by showing "there existed at the time of invention a known problem for which there was an obvious solution encompassed by the patent's claims." *Id.* at 420. "[A]ny need or problem known in the field of endeavor at the time of invention and addressed by the patent can provide a reason for combining the elements in the manner claimed." *Id.* Common sense also teaches that "familiar items may have obvious uses beyond their primary purposes, and in many cases a person of ordinary skill will be able to fit the teachings of multiple patents together like pieces of a puzzle." *Id.*

However, the Supreme Court in *KSR* held that a claimed invention can be obvious even if there is no explicit teaching, suggestion, or motivation for combining the prior art to produce that invention. In summary, *KSR* holds that patents that are based on new combinations of elements or components already known in a technical field may be found to be obvious. *See, generally*, *KSR*, 127 S.Ct. 1727. Specifically, the Court in *KSR* rejected a rigid application of the "teaching, suggestion, or motivation [to combine]" test. *Id.* at 1741. "In determining whether the subject matter of a patent claim is obvious, neither the particular motivation nor the avowed purpose of the patentee controls. What matters is the objective reach of the claim." *Id.* at 1741-1742. "Under the correct analysis, any need or problem known in the field of endeavor at the time of invention and addressed by the patent can provide a reason for combining the elements in the manner claimed." *Id*. at 1742. A key inquiry is whether the "improvement is more than the predictable use of prior art elements according to their established functions." *Id*. at 1740.

The rationale to combine or modify prior art references is significantly stronger when, as here, the references seek to solve the same problem, come from the same field, and correspond well to each other. *In re Inland Steel Co*., 265 F.3d 1354, 1362 (Fed. Cir. 2001). The Federal Circuit has held that two references may be combined as invalidating art under similar circumstances, namely "[the prior art] focus[es] on the same problem that the . . . patent addresses: enhancing the magnetic properties of . . . steel. Moreover, both [prior art references] come from the same field . . . . Finally, the solutions to the identified problems found in the two references correspond well." *Id*. at 1364 (concerning patents and prior art relating to improving the magnetic and electrical properties of steel).

In view of the Supreme Court's *KSR* decision, the PTO issued a set of Examination Guidelines. Examination Guidelines for Determining Obviousness Under 35 U.S.C. §103 in view of the Supreme Court Decision in *KSR International Co. v. Teleflex, Inc*., 72 Fed. Reg. 57526 (October 10, 2007). Those Guidelines summarized the *KSR* decision and identified various rationales for finding a claim obvious, including those based on other precedents. Those rationales include:

> (A) Combining prior art elements according to known methods to yield predictable results;
> (B) Simple substitution of one known element for another to obtain predictable results;
> (C) Use of known technique to improve similar devices (methods, or products) in the same way;
> (D) Applying a known technique to a known device (method, or product) ready for improvement to yield predictable results;
> (E) "Obvious to try" – choosing from a finite number of identified, predictable solutions, with a reasonable expectation of success;

> (F) Known work in one field of endeavor may prompt variations of it for use in either the same field or a different one based on design incentives or other market forces if the variations would have been predictable to one of ordinary skill in the art;
>
> (G) Some teaching, suggestion, or motivation in the prior art that would have led one of ordinary skill to modify the prior art reference or to combine prior art reference teachings to arrive at the claimed invention.

Id. at 57529. The above rationales likewise apply in rendering obvious the asserted claims of the Asserted Patents.

The references disclosed herein, alone or in combination, contain an explicit and/or implicit teaching or motivation to combine them due to the following: (1) the knowledge generally available to a POSITA; (2) the prior art references as understood by a POSITA; (3) the nature of the problem to be solved; (4) the fact that each prior art reference addresses similar problems; and (5) the knowledge of a POSITA that the disclosed elements had been or could be used together.

As an example of those reasons and motivations to combine, ~~the references~~ Cole, Oestvall, Montemurro, Rao, Kelz, Scahill, Wyld, Van Megen, Maes, Vals, Araujo, Freund, Malomsoky, D'Amore, Lee, Brisebois, ~~and~~ Senior, and the various Android and non-Android prior art systems all generally relate to or include network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. *See* Exs. A-01 through I-11; Exs. A-A through I-I. The prior art ~~references~~ depicts, discloses, and discusses similar components and techniques for monitoring network, data, and battery usage, classifying whether applications are running in the foreground or background, and applying differential traffic control policies to save data and lengthen battery life, among other things. *Id.* Thus, a person of ordinary skill in the art would understand the teachings of the references and systems to be applicable to one another. A POSITA would have also found it obvious to implement (i.e., obvious to try) such combinations in order to utilize these well-known data saving, battery saving, and network management techniques in a wireless end-user device capable of communicating data for Internet service activities over both wireless wide area networks and wireless local area networks. *Id.*

For example, a POSITA would look to the primary and secondary references and systems discussed above to improve or tailor the disclosures thereof to help device manufacturers, wireless carriers, and customers reduce data usage and network congestion, extend battery life by decreasing power consumption, and enable users to stay connected. A POSITA would have understood and been aware of motivations to conserve system resources, increase battery life, reduce network congestion, and cut cost by limiting data usage (e.g., by applying a traffic control policy based on whether an application is running in the foreground or background) when communicating over a WWAN.

For example, Android Devices running at least Android versions 1.0-2.2 did this before the alleged inventions.[7] *See* Exs. A-10, B-10, C-10, D-10, E-10, F-10, G-10, H-10, I-10; *see also*, *e.g.*, GOOG-HEADWATER-000000040 ("ConnectivityManager"), *id.* ("Monitor network connections (Wi-Fi, GPRS, UMTS, etc.)"), GOOG-HEADWATER-000000041 ("getNetworkInfo"), *id.* ("TYPE_MOBILE = 0" and "TYPE_WIFI = 1"); GOOG-HEADWATER-000000042 ("Ensure that a network route exists to deliver traffic to the specified host via the specified network interface. An attempt to add a route that already exists is ignored, but treated as successful."); GOOG-HEADWATER-000000035 ("This class gives you control of the power state of the device."); GOOG-HEADWATER-000000001 ("You can use methods on this object to control the power state of the device."); GOOG-HEADWATER-000000097 (noting that "[w]aking up in the

[7] For Android versions 1.0-2.2, code from earlier versions carried over into later versions. Thus, any code cited from an earlier version of Android also exists in the later version.

background when the phone would otherwise be sleeping" "costs the most" battery life); GOOG-HEADWATER-000000101 ("Only update if WiFi or 3G is connected and not roaming"); GOOG-HEADWATER-000000106-11 ("Foreground apps"); GOOG-HEADWATER-000000112-18 ("Background apps"); GOOG-HEADWATER-000000118 ("Background apps . . . Checking current battery and network state before running a full update"), GOOG-HEADWATER-000000026 ("When the user leaves an application, its process is kept around in the background allowing it to continue working . . . if needed, and come immediately to the foreground if the user returns to it."), *id.* ("A Service allows an application to implement longer-running background operations."), *id.* ("The e-mail application schedules an alarm to wake up a service at regular intervals that looks for and retrieves any new mail."); GOOG-HEADWATER-000000032 ("@return Whether background data usage is allowed . . . getBackgroundDataSetting . . . Whether an application should use data while it is in the background."); GOOG-HEADWATER-000000033 ("Sets the persisted value for enabling/disabling Mobile data. . . Whether the mobile data connection should be used or not."); GOOG-HEADWATER-000000029-34 ("If an application uses the network in the background, it should listen for this broadcast and stop using the background data if the value is false. . . If false, applications should not use the network if the application is not in the foreground."); SAMSUNG_PRIORART0005048 ("Monitor network connections (Wi-Fi, GPRS, UMTS, et.c)"); SAMSUNG_PRIORART0005049 ("The setting for background data usage has changed values . . . If an application uses the network in the background, it should listen for this broadcast and stop using the background data if the value is false."); SAMSUNG_PRIORART0005049 ("networkType == TYPE_WIFI || networkType == TYPE_MOBILE"); SAMSUNG_PRIORART0005049 ("TYPE_MOBILE = 0" and "TYPE_WIFI = 1"); SAMSUNG_PRIORART0005050-51 ("Ensure that a network route exists to deliver traffic to the specified host via the specified network interface.”); SAMSUNG_PRIORART0005051 (“Returns the value of the setting for background data usage. If false, applications should not use the network if the application is not in the foreground. Developers should respect this setting, and check the value of this before performing any background data operations.”); SAMSUNG_PRIORART0005051 (“Whether background data usage is allowed . . . Sets the value of the setting for background data usage . . . Whether an application should use data while it is in the background . . . allowBackgroundData”); SAMSUNG_PRIORART0005487 at ConnectivityManager (“Monitor network connections (Wi-Fi, GPRS, UMTS, etc.)”); *id.* at NetworkInfo (“The device is on a network other than the home network (i.e., roaming)”); SAMSUNG_PRIORART0005350 at ConnectivityManager (“Returns the value of the setting for background data usage. If false, applications should not use the network if the application is not in the foreground.”); *id.* at NetworkInfo (“Indicates whether the device is currently roaming on this network. When {@code true}, it suggests that use of data on this network may incur extra costs.”); *id.* at ConnectivityService (“Create the network state trackers for Wi-Fi and mobile data.”); *id.* (“ConnectivityManager#getBackgroundDataSetting”); SAMSUNG_PRIORART0005353 at Fundamentals.jd (“To keep the music going, the media player activity could start a service to run in the background. The system would then keep the music playback service running even after the activity that started it leaves the screen.”); *id.* (“The entire task (the entire activity stack) can be brought to the foreground or sent to the background. . . . It is *active* or *running* when it is in the foreground of the screen (at the top of the activity stack for the current task). This is the activity that is the focus for the user's actions.. . . The foreground lifetime of an activity happens between a call to {@link android.app.Activity#onResume onResume()} until a corresponding call to {@link android.app.Activity#onPause onPause()}.

During this time, the activity is in front of all other activities on screen and is interacting with the user. . . . A foreground process is one that is required for what the user is currently doing. . . A background process is one holding an activity that's not currently visible to the user.”); SAMSUNG_PRIORART0005598 (“Background data . . . Enable background data usage”); *see generally* GOOG-HEADWATER-000000001-123; SAMSUNG_PRIORART0005042-5624.

Accordingly, a POSITA would find these goals and methods for achieving them obvious, would further seek to combine or modify the disclosure of any given primary and secondary references and/or systems to achieve those goals, and would have readily understood that doing so could increase device/network performance, improve user interactions and/or satisfaction, and reduce cost. *See, e.g.*, Oestvall [0009] (“[T]he fact that applications can run in the background and hence still consume some system resource is a waste of CPU and scheduler activity. And in the battery operated, portable device domain, it is especially valuable to conserve system resources wherever possible since doing so can increase battery life[.]”), [0021] (“Preserving system resources could be especially valuable not only in the context of portable, battery powered devices, but also a UPS (uninterruptible power supply) powered system: once activated because a primary power source has ceased to provide power, the need to preserve system resources for as long as possible is very valuable.”); Senior at 3:41-60 (“[O]perating system specific changes are suggested to facilitate bandwidth resource management supplemented with a better design for devices so that devices are able to change their bandwidth requirements dynamically without significantly inconveniencing the users.”), 8:45-55 (“The ability to make such preferences transparently, i.e., without requiring input from the user, is a particularly important benefit made possible in some embodiments. In the rapidly developing Internet environment it is to be expected that a variety of data streams may be accessed via USB devices connected to the Internet or other sources of data.

Transparent operation of the rebalancing-enabler module would give effect to user preferences to give an impression of a tunable seamless experience. Thus, the system would automatically focus on the activities of the greatest interest to the user."); Cole at [0067] ("[A] user may not want to use a 3G network for fast connection because the use may incur added costs per connection."), [0088] ("The user is in a public place and the lengthy transfer is conducted using the WWAN modem 230 (e.g., a 3G channel), which consumes significant battery power."); Montemurro at [0003] ("Selection among the different radio access technologies may be driven by the different properties of the technologies such as bandwidth, range, cost, and power consumption, among other considerations."); Rao at [0003] (teaching that "[i]t is desirable to provide application-aware, client-specific prioritization of packet traffic" to avoid situations where "a network packet generated or received for an application running in the background" is "processed ahead of a network packet generated or received for the application running in the foreground" "and currently in active use by the user"); Wyld at 5:3-19 ("The application can have both background and foreground tasks that access the network. For example, an email program has a foreground task that allows the user to create and then send new messages. While in background, the email program may need to periodically check a remote email server for new messages addressed to the user. In accordance with an embodiment of the invention, performing this checking for new email messages (or, more generally, executing a background task over the network) in response to receiving the signal of network activity idle time, helps make more efficient use of the network capability, thereby improving the user's experience with networking functions on the client device."); Kelz at [0008] (mentioning that its "interplay between the applications, the various protocol stacks supplied by different companies and the wireless link is improved" and, "[a]s a result, the available resources in terms of bandwidth and transmission capacities can be exploited more efficiently"); Scahill at [0010]-[0011] ("By coordinating a plurality of applications in their execution each task can be performed by one of the plurality of applications under suitable network conditions in a way which reduces or removes the likelihood of any conflict with other applications trying to also establish a connection over the same wireless communications link. Even when a wireless communications device can support more than, one network connection it is useful if conflicts can be resolved to ensure applications (and the performance of tasks which require a network connection) are co-ordinated as otherwise conflicts could still occur between applications. It is known in the art to configure a communications enabled device to continually scan for available networks and trigger applications when a particular network is available, particularly in the context of applications registering callbacks for various network interfaces."), [0055]-[0056] ("An application scheduler according to this aspect of the invention system seeks to provide the advantage of only attempting to use a network when the network is expected to be available with sufficient quality and duration to satisfy the application requirements. An application scheduler according to this aspect of the invention also seeks to reduce wastage of device resources, for example, processing power and/or battery power, by eliminating unnecessary scanning operations and network connections. An application scheduler according to the invention also seeks to be capable of operating on a range of communications-enabled devices capable of communicating over a variety of network types, including devices with limited power and minimal processing capabilities such as java enabled feature phones."); Van Megen at 1:6-23 ("Unfortunately, the default network connection may not be the best use of available network resources or may not be in the best interest of the user at the particular time. For example, there may be widely different data transfer rates between available networks and the default selection of a lower bandwidth connection may unduly increase download time or tie up processor capacity. Further, different connection options may have different usage fees and the default network connection may not be the best value to the user at the time."), 2:30-36 ("Because the cost to transfer data overa Wi-Fi connection is generally negligible compared to the cost per data unit (e.g., per megabyte (MB)) transferred over a GPRS connection, it may be desirable to use a Wi-Fi interface whenever possible. The same preference may be true with respect to the available bandwidth, which in general is much larger on a Wi-Fi net work compared to a GPRS connection."), 5:16-26 (noting that binding an application to WLAN prevents "unwanted costs for the user" which would be inccured if the application operated over "a more expensive network connection"); Maes at 1:32-39 ("[A]n access provider may offer a variety of different billing models to customers. Access providers may charge customers based on data traffic, based on services used (which can include access of content), or a combination of data traffic and service usage. With each of these models, the access provider may use a variety of different rating schemes. The rating schemes may be per usage, per levels of usage, per subscription, or combination of these schemes."); Vals at [0018] ("embodiments of the invention are operable … to reduce the power draw of the device thereby reducing infrastructure power costs, among other benefits"); Araujo at [0001] ("Battery-powered machines have a finite amount of energy stored between charges, and these machines generally attempt to use the energy in a way that strikes a balance between providing functionality and maintaining longevity of the charge. Electricity is expensive, so even machines that are connected to power sources manage their use of energy in order to reduce the cost of operation."), [0021] ("since a network card may use more power when transmitting than when idle, a network card … might be directed not to carry out a request in order to conserve power"); Freund at 10:33-37 ("[I]t is desirable to monitor the time an employee spends 'actively' interacting with the Internet, So that management goals of controling counterproductive Web browsing can be realized."), 11:21-28

(“Further, per application monitoring simplifies the task of tracking bandwidth utilization for a network, including providing detailed review on how the Internet access is being used. This greatly eases planning of hardware and connection requirements. Inadvertent disruptions of the network by individual users, such as bandwidth hording by a user using RealAudio for listening to a Web audio ‘broadcast,’ can be averted.”), 27:9-13 (“This allows an administrator, for instance, to Specify that a rule blocking a RealAudio application remains in force during working hours on weekdays-that is, at times when network traffic is already congested.”).

In particular, a POSITA would have been motivated to restrict network access for applications running in the background—especially when communicating over a WWAN or roaming—to conserve power, reduce network congestion, limit the impact such applications could have on applications running in the foreground, and reduce costly data usage associated with data-limited carrier service plans. *See, e.g.*, Oestvall at [0006] (noting that applications that “continue to run even when not actually in active use … will therefore continue to use some system resources, even when residing in the background”), [0024] (“One example use of the present invention is to prevent background … applications from polling for data over a wireless network, an activity that can potentially drain a battery quickly.”); Scahill at [0179] (“Since these applications are also executing effectively autonomously and in the ‘background’ ensuring that one application only runs at one time has the advantage of minimising computer processor unit (CPU) load thereby reducing the impact on any foreground applications that the user is interacting with.”); Montemurro at [0003]-[0004] (“Selection among the different radio access technologies may be driven by the different properties of the technologies such as bandwidth, range, cost, and power consumption, among other considerations…There are costs associated with application access from these different networks.”); Rao at [0003] (teaching that “[i]t is desirable to provide application-aware, client-specific prioritization of packet traffic" to avoid situations where "a network packet generated or received for an application running in the background" is "processed ahead of a network packet generated or received for the application running in the foreground" "and currently in active use by the user"); GOOG-HEADWATER-000000040 ("Monitor network connections (Wi-Fi, GPRS, UMTS, etc.)"), GOOG-HEADWATER-000000041 ("TYPE_MOBILE = 0" and "TYPE_WIFI = 1"); GOOG-HEADWATER-000000035 ("This class gives you control of the power state of the device."); GOOG-HEADWATER-000000101 ("Only update if WiFi or 3G is connected and not roaming"), GOOG-HEADWATER-000000118 ("Background apps . . . Checking current battery and network state before running a full update"), GOOG-HEADWATER-000000026 ("When the user leaves an application, its process is kept around in the background allowing it to continue working . . . if needed, and come immediately to the foreground if the user returns to it."), *id.* ("A Service allows an application to implement longer-running background operations."), *id.* ("The e-mail application schedules an alarm to wake up a service at regular intervals that looks for and retrieves any new mail."); GOOG-HEADWATER-000000001-123; GOOG-HEADWATER-000000029-34 ("If an application uses the network in the background, it should listen for this broadcast and stop using the background data if the value is false. . . If false, applications should not use the network if the application is not in the foreground."); SAMSUNG_PRIORART0005048-51 (same); *see also* Exs. A-01 through I-11; Exs. A-A through I-I.

One of skill in the art would also have been motivated to combine the different publications and patents that were authored by employees of a given company or assigned to the same assignee and/or related to the same subject matter. Additionally, one of skill in the art would have been motivated to combine different references that were authored, developed, or invented by the same individual(s) related to the same subject matter. The common inventor/author/architect of the references demonstrate that they relate to continued work in a common field of effort and continued related developments in that field. One of skill in the art would, therefore, combine the references related to each individual. Additionally, based on the teachings of the references and/or the knowledge of one of ordinary skill, one of skill in the art would have been motivated to combine different references from the same company. For example, a POSITA would have been motivated to combine at least Araujo, Vals, and Van Megen, all of which are currently assigned to the same company (e.g., Microsoft). And, one of skill in the art would have been motivated to combine prior art systems or products (e.g., SymbianOS) with any related or applicable patent or non-patent documentation or literature relating to that system or owned by the same entity (e.g., Oestvall), including for the reason that these materials are related. Similarly, one of skill in the art would have been motivated to combine prior art systems or products (e.g., Windows Mobile Devices, and Windows XP Devices) with any related or applicable patent or non-patent documentation or literature relating to that system or owned by the same entity (e.g., Araujo, Vals, and Van Megen), including for the reason that these materials are related.

Further, below are additional motivations to combine prior art for particular claim limitations. The following discussions of specific claim limitations are merely examples and are not limiting. For example, where a POSITA would have been motivated to combine references which together render obvious limitations from the independent claims, a POSITA would have also been motivated to combine said references in such a way as to render obvious various asserted dependent claims. The motivations identified with respect to any one Asserted Patent apply with equal force to any of the other Asserted Patents by virtue of their relationship and similarities.

a. ’701 patent

To the extent that any primary reference is deemed not to anticipate a claim for failing to teach a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the WWAN, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the WWAN. For example, several prior art references and systems, including at least Cole, Rao, Wyld, Scahill, and Freund, disclose this limitation. *See* Exs. A-01; A-03; A-05; A-06; A-08; A-10; *see also* A-A. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that WWAN connections and modems were routine design choices available for mobile systems as of the critical date. Moreover, WWAN technology was well known as of the critical date and commonly used to connect mobile devices to the internet. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the WWAN. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA’s knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN. For example, several prior art references and systems, including at least Cole, Rao, Wyld, Scahill, and Freund disclose a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN. *See* Exs. A-01; A-03; A-05; A-06; A-08; A-10 *see also* A-A. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that WLAN connections and modems were routine design choices available for mobile systems as of the critical date. Moreover, WLAN modems and technology were well known as of the critical date and commonly used to connect mobile devices to the internet. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the

POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach one or more processors, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses one or more processors. For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Scahill, Wyld, Araujo, Freund, and Senior, disclose one or more processors. *See* Exs. A-01-A-09; A-10; *see also* A-A. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that the inclusion of one or more processers was a routine design choice, commonly implemented in mobile devices as of the critical date. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include one or more processors. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach "one or more processors configured to determine, for a first end-user application . . . , whether the application is running in a background state or as a foreground application," it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses "one or more processors configured to determine, for a first end-user application . . . , whether the application is running in a background state or as a foreground application." For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Wyld, and Freund disclose this limitation. *See* Exs. A-01; A-02; A-03; A-04; A-06; A-08; A-10; *see also* A-A. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would have also understood and been aware of motivations to conserve system resources, increase battery life, reduce network congestion, and cut cost by limiting data usage and/or system resources by determining whether applications running on a mobile device are operating in the foreground or background. Morover, a POSITA would have been motivated to differentiate between foreground and background application processes to efficiently utilize and control the consumption of constrained network bandwidth, processing resources, and/or power consumption, among other system resources. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include "one or more processors configured to determine, for a first end-user application . . . , whether the application is running in a background state or as a foreground application." A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach "one or more processors configured to . . . selectively block and allow access by the first end-user application to the WWAN modem at a time when data for Internet service activities is communicated through a WWAN modem connection to the at least one WWAN, wherein the access is selectively blocked based on a determination that the first end-user application is running in a background state, and wherein the access is selectively allowed based on a determination that the first end-user application is running as a foreground application," it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses "one or more processors configured to . . . selectively block and allow access by the first end-user application to the WWAN modem at a time when data for Internet service activities is communicated through a WWAN modem connection to the at least one WWAN, wherein the access is selectively blocked based on a determination that the first end-user application is running in a background state, and wherein the access is selectively allowed based on a determination that the first end-user application is running as a foreground application." For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Wyld, and Freund, disclose this limitation. *See* Exs. A-01; A-02; A-03; A-04; A-06; A-08; A-10; *see also* A-A. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would have also understood and been aware of motivations to conserve system resources, increase battery life, reduce network congestion, and cut cost by limiting data usage (e.g., by applying a traffic control policy based on whether an application is running in the foreground or background) when communicating over a WWAN. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include "one or more processors configured to . . . selectively block and allow access by the first end-user application to the WWAN modem at a time when data for Internet service activities is communicated through a WWAN modem connection to the at least one WWAN, wherein the access is selectively blocked based on a determination that the first end-user application is running in a background state, and wherein the access is selectively allowed based on a determination that the first end-user application is running as a foreground application." A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate one or more of the dependent claims, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art discussed in Exhibit A-A. For example, Montemurro, Van Megen, Maes, Vals, Malomsoky, D'Amore, Lee, Brisebois, and Aleksic disclose one or more of the asserted dependent claim limitations. *See* Ex. A-A. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would also have been motivated to combine any of these references with any of the primary references for the reasons discussed in Section IV.B.2. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include the features described in Exhibit A-A. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations

b. '184 patent

To the extent that any primary reference is deemed not to anticipate a claim for failing to teach a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the at least one WWAN, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the at least one WWAN. For example, several prior art references and systems, including at least Cole, Rao, Wyld, Scahill, and Freund, disclose this limitation. *See* Exs. B-01; B-03; B-05; B-06; B-08; B-10; *see also* B-B. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that WWAN connections and modems were routine design choices available for mobile systems as of the critical date. Moreover, WWAN technology was well known as of the critical date and commonly used to connect mobile devices to the internet. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the at least one WWAN. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach one or more processors, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses one or more processors. For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Scahill, Wyld, Araujo, Freund, and Senior, disclose one or more processors. *See* Exs. B-01-B-09; B-10; *see also* B-B. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that the inclusion of one or more processers was a routine design choice, commonly implemented in mobile devices as of the critical date. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include one or more processors. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach "one or more processors configured to . . . classify whether a particular application associated with an Internet service access request is interacting with the user in the device user interface foreground," it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses "one or more processors configured to . . . classify whether a particular application associated with an Internet service access request is interacting with the user in the device user interface foreground." For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Wyld, and Freund disclose this limitation. *See* Exs. B-01; B-02; B-03; B-04; B-06; B-08; B-10; *see also* B-B. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would have also understood and been aware of motivations to conserve system resources, increase battery life, reduce network congestion, and cut cost by limiting data usage and/or system resources by determining whether applications running on a mobile device are operating in the foreground or background. Morover, a POSITA would have been motivated to differentiate between foreground and background application processes to efficiently utilize and control the consumption of constrained network bandwidth, processing resources, and/or power consumption, among other system resources. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include "one or more processors configured to . . . classify whether a particular application associated with an Internet service access request is interacting with the user in the device user interface foreground." A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach "one or more processors configured to, for a time when data communication for Internet service activities is provided by the WWAN modem . . . apply a differential traffic control policy to the Internet service access request, based on . . . whether the application is classified as interacting with the user," it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses "one or more processors configured to, for a time when data communication for Internet service activities is provided by the WWAN modem . . . apply a differential traffic control policy to the Internet service access request, based on . . . whether the application is classified as interacting with the user." For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Wyld, and Freund, disclose this limitation. *See* Exs. B-01; B-02; B-03; B-04; B-06; B-08; B-10; *see also* B-B. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would have also understood and been aware of motivations to conserve system resources, increase battery life, reduce network congestion, and cut cost by limiting data usage (e.g., by applying a traffic control policy based on whether an application is running in the foreground or background) when communicating over a WWAN. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include "one or more processors configured to, for a time when data communication for Internet service activities is provided by the WWAN modem . . . apply a differential traffic control policy to the Internet service access request, based on . . . whether the application is classified as interacting with the user." A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach for at least one of the first or second one or more applications, disallow Internet data communication for that application based on an application-specific amount of Internet data usage reaching a limit, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses for at least one of the first or second one or more applications, disallow Internet data communication for that application based on an application-specific amount of Internet data usage reaching a limit. For example, several prior art references and systems, including at least Oestval, Rao, Kelz, Scahill, and Freund, disclose this limitation. *See* Exs. B-02; B-03; B-04; B-05; B-08; B-10; *see also* B-B. It would have been obvious to a person skilled in the art to incorporate such features to meet the shared goal of efficiently managing scarce resources (*e.g.*, battery life and/or data). For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. *See, e.g.*, Oestvall at [0004] ("Battery conservation in battery operated computing devices is very important[.]"). Moreover, limiting data usage to conserve system resources in a wireless communication device was well known in the art at least as of the critical date. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include for at least one of the first or second one or more applications, disallow Internet data communication for that application based on an application-specific amount of Internet data usage reaching a limit. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate one or more of the dependent claims, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art discussed in Exhibit B-B. For example, Montemurro, Van Megen, Maes, Vals, Malomsoky, D'Amore, Lee, Brisebois, and

Aleksic disclose one or more of the asserted dependent claim limitations. *See* Ex. B-B. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would also have been motivated to combine any of these references with any of the primary references for the reasons discussed in Section IV.B.2. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include the features described in Exhibit B-B. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

c. '578 patent

To the extent that any primary reference is deemed not to anticipate a claim for failing to teach a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the at least one WWAN, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the at least one WWAN. For example, several prior art references and systems, including at least Cole, Rao, Wyld, Scahill, and Freund, disclose a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the at least one WWAN. *See* Exs. C-01; C-03; C-05; C-06; C-08; C-10; *see also* C-C. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that WWAN connections and modems were routine design choices available for mobile systems as of the critical date. Moreover, WWAN technology was well known as of the critical date and commonly used to connect mobile devices to the internet. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the at least one WWAN. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the at least one WLAN, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the at least one WLAN. For example, several prior art references and systems, including at least Cole, Rao, Wyld, Scahill, and Freund disclose a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the at least one WLAN. *See* Exs. C-01; C-03; C-05; C-06; C-08; C-10; *see also* C-C. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that WLAN connections and modems were routine design choices available for mobile systems as of the critical date. Moreover, WLAN modems and technology were well known as of the critical date and commonly used to connect mobile devices to the internet. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the at least one WLAN. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach one or more processors, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses one or more processors. For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Scahill, Wyld, Araujo, Freund, and Senior, disclose one or more processors. *See* Exs. C-01-C-09; C-10; *see also* C-C. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that the inclusion of one or more processers was a routine design choice, commonly implemented in mobile devices as of the critical date. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include one or more processors. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach "one or more processors configured to implement an application program interface (API) that allows a particular application to access one or more aspects of the differential traffic control policy applicable to that application, including whether the user-settable aspects of the policy only allow the particular application to utilize the at least one WWAN for Internet service activities when the particular application is classified as interacting with a user in the device user interface foreground," it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses "one or more processors configured to implement an application program interface (API) that allows a particular application to access one or more aspects of the differential traffic control policy applicable to that application, including whether the user-settable aspects of the policy only allow the particular application to utilize the at least one WWAN for Internet service activities when the particular application is classified as interacting with a user in the device user interface foreground." For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Wyld, and Freund, disclose this limitation. *See* Exs. C-01; C-02; C-03; C-04; C-06; C-08; C-10; *see also* C-C. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would have also understood and been aware of motivations to conserve system resources, increase battery life, reduce network congestion, and cut cost by limiting data usage (e.g., by applying a traffic control policy based on whether an application is running in the foreground or background) when communicating over a WWAN. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include "one or more processors configured to implement an application program interface (API) that allows a particular application to access one or more aspects of the differential traffic control policy applicable to that application, including whether the user-settable aspects of the policy only allow the particular application to utilize the at least one WWAN for Internet service activities when the particular application is classified as interacting with a user in the device user interface foreground." A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach "one or more processors configured to implement an application program interface (API) that allows a particular application to access one or more aspects of the differential traffic control policy applicable to that application . . . when the particular application is classified as interacting with a user in the device user interface foreground," it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses "one or more processors configured to implement an application program interface (API) that allows a particular application to access one or more aspects of the differential traffic control policy applicable to that application . . . when the particular application is classified as interacting with a user in the device user interface foreground." For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Wyld, and Freund disclose this limitation. *See* Exs. C-01; C-02; C-03; C-04; C-06; C-08; C-10; *see also* C-C. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would have also understood and been aware of motivations to conserve system resources, increase battery life, reduce network congestion, and cut cost by limiting data usage and/or system resources by determining whether applications running on a mobile device are operating in the foreground or background. Morover, a POSITA would have been motivated to differentiate between foreground and background application processes to efficiently utilize and control the consumption of constrained network bandwidth, processing resources, and/or power consumption, among other system resources. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include "one or more processors configured to implement an application program interface (API) that allows a particular application to access one or more aspects of the differential traffic control policy applicable to that application . . . when the particular application is classified as interacting with a user in the device user interface foreground." A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate one or more of the dependent claims, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art discussed in Exhibit C-C. For example, Montemurro, Van Megen, Maes, Vals, Malomsoky, D'Amore, Lee, Brisebois, and Aleksic disclose one or more of the asserted dependent claim limitations. *See* Ex. C-C. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would also have been motivated to combine any of these references with any of the primary references for the reasons discussed in Section IV.B.2. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include the features described in Exhibit C-C. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

d. '445 patent

To the extent that any primary reference is deemed not to anticipate a claim for failing to teach a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the WWAN, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the WWAN. For example, several prior art references and systems, including at least Cole, Rao, Wyld, Scahill, and Freund, disclose a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the WWAN. *See* Exs. D-01; D-03; D-05; D-06; D-08; D-10; *see also* D-D. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that WWAN connections and modems were routine design choices available for mobile systems as of the critical date. Moreover, WWAN technology was well known as of the critical date and commonly used to connect mobile devices to the internet. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the WWAN. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN. For example, several prior art references and systems, including at least Cole, Rao, Wyld, Scahill, and Freund disclose a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN. *See* Exs. D-01; D-03; D-05; D-06; D-08; D-10; *see also* D-D. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that WLAN connections and modems were routine design choices available for mobile systems as of the critical date. Moreover, WLAN modems and technology were well known as of the critical date and commonly used to connect mobile devices to the internet. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach one or more processors, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses one or more processors. For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Scahill, Wyld, Araujo, Freund, and Senior, disclose one or more processors. *See* Exs. D-01-D-09; D-10; *see also* D-D. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that the inclusion of one or more processers was a routine design choice, commonly implemented in mobile devices as of the critical date. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include one or more processors. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach "one or more processors configured to . . . classify . . . , for a first end-user application . . . , whether a particular application associated with an Internet service access request . . . is interacting with the user in the device user interface foreground," it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses "one or more processors configured to . . . classify . . . , for a first end-user application . . . , whether a particular application associated with an Internet service access request . . . is interacting with the user in the device user interface foreground." For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Wyld, and Freund disclose this limitation. *See* Exs. D-01; D-02; D-03; D-04; D-06; D-08; D-10; *see also* D-D. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would have also understood and been aware of motivations to conserve system resources, increase battery life, reduce network congestion, and cut cost by limiting data usage and/or system resources by determining whether applications running on a mobile device are operating in the foreground or background. Morover, a POSITA would have been motivated to differentiate between foreground and background application processes to efficiently utilize and control the consumption of constrained network bandwidth, processing resources, and/or power consumption, among other system resources. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include "one or more processors configured to . . . classify . . . , for a first end-user application . . . , whether a particular application associated with an Internet service access request . . . is interacting with the user in the device user interface foreground." A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach "one or more processors configured to . . . apply a differential traffic control policy to the Internet service access request, based on [whether data for nternet service activities is to be communicated through the WWAN modem or the WLAN modem and whether a particular application . . . is interacting with the user in the device user interface foreground]," it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses "one or more processors configured to . . . apply a differential traffic control policy to the Internet service access request, based on [whether data for nternet service activities is to be communicated through the WWAN modem or the WLAN modem and whether a particular application . . . is interacting with the user in the device user interface foreground]." For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Wyld, and Freund, disclose this limitation. *See* Exs. D-01; D-02; D-03; D-04; D-06; D-08; D-10; *see also* D-D. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would have also understood and been aware of motivations to conserve system resources, increase battery life, reduce network congestion, and cut cost by limiting data usage (e.g., by applying a traffic control policy based on whether an application is running in the foreground or background) when communicating over a WWAN. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include "one or more processors configured to . . . apply a differential traffic control policy to the Internet service access request, based on [whether data for nternet service activities is to be communicated through the WWAN modem or the WLAN modem and whether a particular application . . . is interacting with the user in the device user interface foreground]." A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the

POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate one or more of the dependent claims, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art discussed in Exhibit D-D. For example, Montemurro, Van Megen, Maes, Vals, Malomsoky, D'Amore, Lee, Brisebois, and Aleksic disclose one or more of the asserted dependent claim limitations. *See* Ex. D-D. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would also have been motivated to combine any of these references with any of the primary references for the reasons discussed in Section IV.B.2. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include the features described in Exhibit D-D. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

e. '224 patent

To the extent that any primary reference is deemed not to anticipate a claim for failing to teach at least one wireless modem, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses at least one wireless modem. For example, several prior art references and systems, including at least Cole, Rao, Wyld, Scahill, and Freund, disclose at least one wireless modem. *See* Exs. E-01; E-03; E-05; E-06; E-08; E-10; *see also* E-E. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that wireless connections and modems were routine design choices available for mobile systems as of the critical date. Moreover, wireless technology was well known as of the critical date and commonly used to connect mobile devices to the internet. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include at least one wireless modem. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach a processor, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses one or more processors. For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Scahill, Wyld, Araujo, Freund, and Senior, disclose a processor. *See* Exs. E-01-E-09; E-10; *see also* E-E. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that the inclusion of a processor was a routine design choice, commonly implemented in mobile devices as of the critical date. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a processor. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach for each given application of a plurality of applications on the wireless end-user device, monitor a network service usage activity of the wireless end-user communications device associated with the given application, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses for each given application of a plurality of applications on the wireless end-user device, monitor a network service usage activity of the wireless end-user communications device associated with the given application. For example, several prior art references and systems, including at least Cole, Oestvall, Kelz, Scahill, and Wyld, disclose for each given application of a plurality of applications on the wireless end-user device, monitor a network service usage activity of the wireless end-user communications device associated with the given application. *See* Exs. E-01; E-02; E-04; E-05; E-06; E-10; *see also* E-E. It would have been obvious to a person skilled in the art to incorporate such functionality, at least because these references shared the goal of efficiently utilizing system resources by prioritizing certain resource-intensive activity (*e.g.*, applications accessing a wireless network). Moreover, it was well known in the art to utilize network monitoring techniques (such as those disclosed in Cole, Oestvall, Kelz, Scahill, and/or Wyld) to optimize system resource usage. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include for each given application of a plurality of applications on the wireless end-user device, monitor a network service usage activity of the wireless end-user communications device associated with the given application. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach "a processor configured to . . . manage network data access via the at least one wireless modem for each of the plurality of applications according to the dynamically selected service usage control policy for each given network service activity of that application," it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses "a processor configured to . . . manage network data access via the at least one wireless modem for each of the plurality of applications according to the dynamically selected service usage control policy for each given network service activity of that application." For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Wyld, and Freund, disclose this limitation. *See* Exs. E-01; E-02; E-03; E-04; E-06; E-08; E-10; *see also* E-E. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would have also understood and been aware of motivations to conserve system resources, increase battery life, reduce network congestion, and cut cost by limiting data usage (e.g., by managing network access) when communicating over a wireless network. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include "a processor configured to . . . manage network data access via the at least one wireless modem for each of the plurality of applications according to the dynamically selected service usage control policy for each given network service activity of that application." A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach a memory coupled to the processor and configured to provide the processor with instructions, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a memory coupled to the processor and configured to provide the processor with instructions. For example, several prior art references and systems, including at least Cole, Rao, Scahill, Wyld, Araujo, Freund, and Senior, disclose a memory coupled to the processor and configured to provide the processor with instructions. *See* Exs. E-01; E-03; E-05; E-06; E-07; E-08; E-09; E-10; *see also* E-E. It would have been obvious to a person skilled in the art to incorporate such features at least because each of these references relate generally to computing devices, for which a POSITA would understand that memory was a necessary constituent. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a memory coupled to the processor and configured to provide the processor with instructions. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate one or more of the dependent claims, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art discussed in Exhibit E-E. For example, Montemurro, Van Megen, Maes, Vals, Malomsoky, D'Amore, Lee, Brisebois, and Aleksic disclose one or more of the asserted dependent claim limitations. *See* Ex. E-E. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would also have been motivated to combine any of these references with any of the primary references for the reasons discussed in Section IV.B.2. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include the features described in Exhibit E-E. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

f. '976 patent

To the extent that any primary reference is deemed not to anticipate a claim for failing to teach a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the WWAN, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the WWAN. For example, several prior art references and systems, including at least Cole, Rao, Wyld, Scahill, and Freund, disclose a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the WWAN. *See* Exs. F-01; F-03; F-05; F-06; F-08; F-10; *see also* F-F. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that WWAN connections and modems were routine design choices available for mobile systems as of the critical date. Moreover, WWAN technology was well known as of the critical date and commonly used to connect mobile devices to the internet. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the WWAN. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN. For example, several prior art references and systems, including at least Cole, Rao, Wyld, Scahill, and Freund disclose a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN. *See* Exs. F-01; F-03; F-05; F-06; F-08; F-10; *see also* F-F. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that WLAN connections and modems were routine design choices available for mobile systems as of the critical date. Moreover, WLAN modems and technology were well known as of the critical date and commonly used to connect mobile devices to the internet. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach one or more processors, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses one or more processors. For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Scahill, Wyld, Araujo, Freund, and Senior, disclose one or more processors. *See* Exs. F-01-F-09; F-10; *see also* F-F. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that the inclusion of one or more processers was a routine design choice, commonly implemented in mobile devices as of the critical date. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include one or more processors. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach "one or more processors configured to classify, for a first end-user application . . . , whether or not the first end-user application, when running, is interacting in the device display foreground with the user," it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses "one or more processors configured to classify, for a first end-user application . . . , whether or not the first end-user application, when running, is interacting in the device display foreground with the user." For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Wyld, and Freund disclose this limitation. *See* Exs. F-01; F-02; F-03; F-04; F-06; F-08; F-10; *see also* F-F. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would have also understood and been aware of motivations to conserve system resources, increase battery life, reduce network congestion, and cut cost by limiting data usage and/or system resources by determining whether applications running on a mobile device are operating in the foreground or background. Morover, a POSITA would have been motivated to differentiate between foreground and background application processes to efficiently utilize and control the consumption of constrained network bandwidth, processing resources, and/or power consumption, among other system resources. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include "one or more processors configured to classify, for a first end-user application . . . , whether or not the first end-user application, when running, is interacting in the device display foreground with the user." A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach "a processor configured to . . . for a time period when data for Internet service activities is communicated through a WWAN modem connection to the at least one WWAN, apply a first differential traffic control policy to Internet service activity on behalf of the first end-user application, such that Internet service activity on behalf of the first end-user application is disallowed when the one or more processors classify the first end-user application as not interacting in the device display foreground with the user," it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses "a processor configured to . . . for a time period when data for Internet service activities is communicated through a WWAN modem connection to the at least one WWAN, apply a first differential traffic control policy to Internet service activity on behalf of the first end-user application, such that Internet service activity on behalf of the first end-user application is disallowed when the one or more processors classify the first end-user application as not interacting in the device display foreground with the user." For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Wyld, and Freund, disclose this limitation. *See* Exs. F-01; F-02; F-03; F-04; F-06; F-08; F-10; *see also* F-F. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would have also understood and been aware of motivations to conserve system resources, increase battery life, reduce network congestion, and cut cost by limiting data usage (e.g., by applying a traffic control policy based on whether an application is running in the foreground or background) when communicating over a WWAN. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include "a processor configured to . . . for a time period when data for Internet service activities is communicated through a WWAN modem connection to the at least one WWAN, apply a first differential traffic control policy to Internet service activity on behalf of the first end-user application, such that Internet service activity on behalf of the first end-user application is disallowed when the one or more processors classify the first end-user application as not interacting in the device display foreground with the user." A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate one or more of the dependent claims, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art discussed in Exhibit F-F. For example, Montemurro, Van Megen, Maes, Vals, Malomsoky, D'Amore, Lee, Brisebois, and Aleksic disclose one or more of the asserted dependent claim limitations. *See* Ex. F-F. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would also have been motivated to combine any of these references with any of the primary references for the reasons discussed in Section IV.B.2. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include the features described in Exhibit F-F. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

g. '433 patent

To the extent that any primary reference is deemed not to anticipate a claim for failing to teach a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the at least one WWAN, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the at least one WWAN, the at least one WWAN having a corresponding network type of a plurality of wireless network types supported by the device for data communication. For example, several prior art references and systems, including at least Cole, Rao, Wyld, Scahill, and Freund, disclose a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the at least one WWAN. *See* Exs. G-01; G-03; G-05; G-06; G-08; G-10; *see also* G-G. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that WWAN connections and modems were routine design choices available for mobile systems as of the critical date. Moreover, WWAN technology was well known as of the critical date and commonly used to connect mobile devices to the internet. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the at least one WWAN. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the

POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach a non-transitory memory to store a network service activity control policy set, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a non-transitory memory to store a network service activity control policy set. For example, several prior art references and systems, including at least Cole, Rao, Scahill, Wyld, Araujo, Freund, and Senior, disclose a memory coupled to the processor and configured to provide the processor with instructions. *See* Exs. G-01; G-03; G-05; G-06; G-07; G-08; G-09; G-10; *see also* G-G. It would have been obvious to a person skilled in the art to incorporate such features at least because each of these references relate generally to computing devices, for which a POSITA would understand that memory was a necessary constituent. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a non-transitory memory to store a network service activity control policy set. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach one or more processors, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses one or more processors. For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Scahill, Wyld, Araujo, Freund, and Senior, disclose one or more processors.

See Exs. G-01-G-09; G-10; *see also* G-G. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that the inclusion of one or more processers was a routine design choice, commonly implemented in mobile devices as of the critical date. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include one or more processors. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate one or more of the dependent claims, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art discussed in Exhibit G-G. For example, Montemurro, Van Megen, Maes, Vals, Malomsoky, D'Amore, Lee, Brisebois, and Aleksic disclose one or more of the asserted dependent claim limitations. *See* Ex. G-G. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would also have been motivated to combine any of these references with any of the primary references for the reasons discussed in Section IV.B.2. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include the features described in Exhibit G-G. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

h. '544 patent

To the extent that any primary reference is deemed not to anticipate a claim for failing to teach a wireless modem to communicate data for network service usage activities between the device and a wireless network, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a wireless modem to communicate data for network service usage activities between the device and a wireless network. For example, several prior art references and systems, including at least Cole, Rao, Wyld, Scahill, and Freund, disclose a wireless modem to communicate data for network service usage activities between the device and a wireless network. *See* Exs. H-01; H-03; H-05; H-06; H-08; H-10; *see also* H-H. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that WWAN connections and modems were routine design choices available for mobile systems as of the critical date. Moreover, WWAN technology was well known as of the critical date and commonly used to connect mobile devices to the internet. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a wireless modem to communicate data for network service usage activities between the device and a wireless network. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach one or more processors, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses one or more processors. For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Scahill, Wyld, Araujo, Freund, and Senior, disclose one or more processors. *See* Exs. H-01-H-09; H-10; *see also* H-H. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that the inclusion of one or more processers was a routine design choice, commonly implemented in mobile devices as of the critical date. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include one or more processors. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach "a processor that executes instructions to associate network service usage activity, on behalf of a first device application, and that occurs when the first device application is not in the foreground of user interaction, with a network service usage control policy," it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a processor that executes instructions to associate network service usage activity, on behalf of a first device application, and that occurs when the first device application is not in the foreground of user interaction, with a network service usage control policy." For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Wyld, and Freund disclose this limitation. *See* Exs. H-01; H-02; H-03; H-04; H-06; H-08; H-10; *see also* H-H. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would have also understood and been aware of motivations to conserve system resources, increase battery life, reduce network congestion, and cut cost by limiting data usage and/or system resources by determining whether applications running on a mobile device are operating in the foreground or background. Morover, a POSITA would have been motivated to differentiate between foreground and background application processes to efficiently utilize and control the consumption of constrained network bandwidth, processing resources, and/or power consumption, among other system resources. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a processor that executes instructions to associate network service usage activity, on behalf of a first device application, and that occurs when the first device application is not in the foreground of user interaction, with a network service usage control policy." A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach "a processor that executes instructions to . . . dynamically determine whether to apply the network service usage control policy to the particular network service usage activity, based on the application state and based on a power control state," it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses "a processor that executes instructions to . . . dynamically determine whether to apply the network service usage control policy to the particular network service usage activity, based on the application state and based on a power control state." For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Wyld, Freund, and Vals, disclose this limitation. *See* Exs. H-01; H-02; H-03; H-04; H-06; H-08; H-10; H-H. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would have also understood and been aware of motivations to conserve system resources, increase battery life, reduce network congestion, and cut cost by limiting data usage (e.g., by applying a traffic control policy based on whether an application is running in the foreground or background and/or a power control state of the device) when communicating over a wireless network. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include "a processor that executes instructions to . . . dynamically determine whether to apply the network service usage control policy to the particular network service usage activity, based on the application state and based on a power control state." A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach a memory coupled to the processor to provide the processor with the instructions, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a memory coupled to the processor to provide the processor with the instructions. For example, several prior art references and systems, including at least Cole, Rao, Scahill, Wyld, Araujo, Freund, and Senior, disclose a memory coupled to the processor and configured to provide the processor with instructions. *See* Exs. H-01; H-03; H-05; H-06; H-07; H-08; H-09; H-10; *see also* H-H. It would have been obvious to a person skilled in the art to incorporate such features at least because each of these references relate generally to computing devices, for which a POSITA would understand that memory was a necessary constituent. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a memory coupled to the processor to provide the processor with the instructions. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach "dynamically assigning a network capacity controlled services priority level to the particular network service usage based on a network busy state," it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses "dynamically assigning a network capacity controlled services priority level to the particular network service usage based on a network busy state." For example, several prior art references, including at least Lee, Montemurro, and Aleksic, disclose a memory coupled to the processor and configured to provide the processor with instructions. *See* Ex. H-H. It would have been obvious to a person skilled in the art to incorporate such features at least because each of these references relate generally to optimizing computing device resources and network utilization. For example, a POSITA would have found it obvious to modify any of the primary references to include such features at least because such features improve network bandwidth usage efficiency and, in turn, reduce system resource requirements and/or more efficiently allocate those resources. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include "dynamically assigning a network capacity controlled services priority level to the particular network service usage based on a network busy state." A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate one or more of the dependent claims, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art discussed in Exhibit H-H. For example, Montemurro, Van Megen, Maes, Vals, Malomsoky, D'Amore, Lee, Brisebois, and Aleksic disclose one or more of the asserted dependent claim limitations. *See* Ex. H-H. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would also have been motivated to combine any of these references with any of the primary references for the reasons discussed in Section IV.B.2. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include the features described in Exhibit H-H. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

i. ’773 patent

To the extent that any primary reference is deemed not to anticipate a claim for failing to teach a processor, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses one or more processors. For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Scahill, Wyld, Araujo, Freund, and Senior, disclose a processor. *See* Exs. I-01-I-10~~09~~; *see also* I-I. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that the inclusion of a processor was a routine design choice, commonly implemented in mobile devices as of the critical date. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a processor. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA’s knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach a wireless modem to communicate data for network service usage activities between the device and a wireless network, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a wireless modem to communicate data for network service usage activities between the device and a wireless network. For example, several prior art references and systems, including at least Cole, Rao, Wyld, Scahill, and Freund, disclose a wireless modem to communicate data for network service usage activities between the device and a wireless network. *See* Exs. I-01; I-03; I-05; I-06; I-08; I-10; *see also* I-I. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would understand that wireless connections and modems were routine design choices available for mobile systems as of the critical date. Moreover, wireless technology was well known as of the critical date and commonly used to connect mobile devices to the internet. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a wireless modem to communicate data for network service usage activities between the device and a wireless network. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach a non-transitory computer-readable storage medium storing instructions, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses a non-transitory computer-readable storage medium storing instructions. For example, several prior art references and systems, including at least Cole, Rao, Scahill, Wyld, Araujo, Freund, and Senior, disclose a memory coupled to the processor and configured to provide the processor with instructions. *See* Exs. I-01; I-03; I-05; I-06; I-07; I-08; I-09; I-10; *see also* I-I. It would have been obvious to a person skilled in the art to incorporate such features at least because each of these references relate generally to computing devices, for which a POSITA would understand that memory was a necessary constituent. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include a non-transitory memory to store a network service activity control policy set. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach "a non-transitory computer-readable storage medium storing instructions that, when provided to the processor, cause the processor to . . . select a corresponding current service usage control policy for the background network service usage activities," it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses "a non-transitory computer-readable storage medium storing instructions that, when provided to the processor, cause the processor to . . . select a corresponding current service usage control policy for the background network service usage activities." For example, several prior art references and systems, including at least Cole, Oestvall, Rao, Kelz, Wyld, and Freund disclose this limitation. *See* Exs. I-01; I-02; I-03; I-04; I-06; I-08; I-10; *see also* I-I. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would have also understood and been aware of motivations to conserve system resources, increase battery life, reduce network congestion, and cut cost by limiting data usage and/or system resources by determining whether applications running on a mobile device are operating in the foreground or background. Morover, a POSITA would have been motivated to differentiate between foreground and background application processes to efficiently utilize and control the consumption of constrained network bandwidth, processing resources, and/or power consumption, among other system resources. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include "a non-transitory computer-readable storage medium storing instructions that, when provided to the processor, cause the processor to . . . select a corresponding current service usage control policy for the background network service usage activities." A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach allow each of the background network service usage activities to access the WWAN during the respective deferred time slot for that background network service usage activity, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses allow each of the background network service usage activities to access the WWAN during the respective deferred time slot for that background network service usage activity. For example, several prior art references and systems, including at least Cole, Oestvall, and Scahill, disclose this limitation. *See* Exs. I-01; I-02; I-05; I-10; *see also* I-I. It would have been obvious to a person skilled in the art to incorporate such features into any of the primary references. For, example, it would have been obvious to implement techniques for optimized system resource usage/allocation (such as those disclosed in Cole, Oestvall, and/or Scahill) into the device disclosed in any primary reference. It was known in the art that wirelss communications devices require optimized resource usage because of their limited power (e.g., battery). Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include allow each of the background network service usage activities to access the WWAN during the respective deferred time slot for that background network service usage activity. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate a claim for failing to teach service usage policy application based on "a network busy state," it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art that discloses service usage policy application based on "a network busy state." For example, several prior art references, including at least Lee, Montemurro, and Aleksic, disclose a memory coupled to the processor and configured to provide the processor with instructions. *See* Ex. I-I; I-10. It would have been obvious to a person skilled in the art to incorporate such features at least because each of these references relate generally to optimizing computing device resources and network utilization. For example, a POSITA would have found it obvious to modify any of the primary references to include such features at least because such features improve network bandwidth usage efficiency and, in turn, reduce system resource requirements and/or more efficiently allocate those resources. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include service usage policy application based on "a network busy state." A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

As another example, to the extent that any primary reference is deemed not to anticipate one or more of the dependent claims, it would have been obvious to a POSITA at the time of the invention to combine the primary reference with any of the prior art discussed in Exhibit I-I. For example, Montemurro, Van Megen, Maes, Vals, Malomsoky, D'Amore, Lee, Brisebois, and Aleksic disclose one or more of the asserted dependent claim limitations. *See* Ex. I-I. It would have been obvious to a person skilled in the art to incorporate such functionality, components, and/or features. For example, a POSITA would understand that each of these references generally relate to network data saving features, battery saving features, and network connectivity management functions, and thus constitute analogous art within the same field of endeavor. In addition, a POSITA would be motivated to incorporate such functionality in order to improve device performance, improve battery life and efficiency, and reduce costs associated with data usage. A POSITA would also have been motivated to combine any of these references with any of the primary references for the reasons discussed in Section IV.B.2. Thus, design and market forces would have motivated a POSITA to modify any of the primary references to include the features described in Exhibit I-I. A POSITA would also have had a reasonable expectation of success in making such modifications to any primary reference. A POSITA would have understood that these references, as well as the POSITA's knowledge, disclose interrelated teachings based on routine technologies and would have been amenable to various well-understood and predictable combinations.

3. Lack of Secondary Indicia of Nonobviousness

Samsung is not aware of any evidence that would tend to establish any secondary considerations of non-obviousness. This lack of evidence further renders the Asserted Claims obvious. Proving any such secondary considerations is Headwater's burden. *See, e.g.*, *ZUP, LLC v. Nach Mfg., Inc.*, 896 F.3d 1365, 1373 (Fed. Cir. 2018) ("[A] patentee bears the burden of production with respect to evidence of secondary considerations of nonobviousness."). Accordingly, Samsung reserves all rights regarding its full contention in this respect until after Headwater completes its final and binding disclosure of any such evidence and contentions. In the meantime, Samsung note the complete lack of any such evidence in the record.

Headwater has disclosed no evidence of, and Samsung knows of no viable evidence to suggest:

- **The alleged invention's commercial success**. Indeed, no products are known to practice the Asserted Claims. To the extent Headwater asserts that Samsung's products practice the Asserted Patents, Samsung denies that assertion and incorporates its responses to date and any future contentions, expert reports, and testimony. Further, Samsung knows of no nexus between any commercial success and the Asserted Claims.

See, e.g., *Windsurfing Int'l Inc. v. AMF* , 782 F.2d 995 (Fed. Cir. 1986) (considerations such as intervening, non-covered technological innovations, popularity of accessories, and advertising expense are all relevant to the nexus determination). If any commercial success is due to any of the concepts discussed in the Asserted Patents, those concepts are also present in the prior art, as described above, and thus do not support any commercial success that is relevant to the question of obviousness. *See Tokai Corp. v. Easton Enters, Inc.* , 632 F.3d 1358, 1369–70 (Fed. Cir. 2011) ("If commercial success is due to an element in the prior art, no nexus exists."); *In re Huai-Hung Kao* , 639 F.3d 1057, 1068 (Fed. Cir. 2011) ("Where the offered secondary consideration actually results from something other than what is both claimed and *novel* in the claim, there is no nexus to the merits of the claimed invention."); *Ormco Corp. v. Align Tech., Inc.* , 463 F.3d 1299, 1312 (Fed. Cir. 2006) ("[I]f the feature that creates the commercial success was known in the prior art, the success is not pertinent.").

- **Alleged commercial success via licensning**. Headwater has presented no evidence of commercial success via a licensing program.
- **Long felt but unresolved needs**. Headwater has presented no evidence of any long felt and unresolved need.
- **No industry praise.** There is also no evidence of industry praise for the alleged invention of the Asserted Patents or any functionality that allegedly practices the Asserted Patents. To the extent any praise is related to any functionality that allegedly practices the Asserted Patents, that praise is not due to the allegedly novel features of the Asserted Patents, but instead only to features present in the prior art, which is not a sufficient nexus to be relevant to the question of industry praise for purposes of

obviousness. *See Muniauction, Inc. v. Thomson Corp.*, 532 F.3d 1318, 1328 (Fed. Cir. 2008). Praise of Samsung's mobile phones or of certain Google Android features is not praise of the Asserted Patents.

- **Unexpected results**: No evidence of any such unexpected results is known. As discussed above, the concepts contained in the Asserted Claims were already combined in the same manner as the asserted. These prior art systems, as described in the above-referenced exhibits, disclosed the same combination of elements, and the same result of that combination, that is recited in the claim. Thus, there were no unexpected results that arose from combining the well-known elements in the Asserted Claims.
- **The failure of others**. No evidence of any such failure is known.
- **Skepticism by experts**. No experts or person of skill expressed skepticism about implementing the alleged inventions.
- **Teaching away by others**. No evidence of any such teaching is known.
- **Recognition of a problem**. As discussed above, the industry recognized the problem and had already discussed multiple approaches that implemented the Asserted Claims to solve that problem.
- **Copying of the alleged invention by competitors**. No evidence of any such copying is known. *See Amazon.com, Inc. v. Barnesandnoble.com, Inc.*, 239 F.3d 1343, 1366 (Fed. Cir. 2001) (allegedly copied feature must be an embodiment of the patented claims).

V. OBVIOUSNESS-TYPE DOUBLE PATENTING

Each of the claims identified below is invalid due to obviousness-type double patenting.

As the Federal Circuit discussed in *In re Hubbell*, obviousness-type double patenting has two separate and independent rationales:

> There are two justifications for obviousness-type double patenting. The first is "to prevent unjustified timewise extension of the right to exclude granted by a patent no matter how the extension is brought about." *Van Ornum*, 686 F.2d at 943-44 (quotation and citation omitted). The second rationale is to prevent multiple infringement suits by different assignees asserting essentially the same patented invention. *Fallaux*, 564 F.3d at 1319 (recognizing that "harassment by multiple assignees" provides "a second justification for obviousness-type double patenting"); *see also* Chisum on Patents § 9.04[2][b][ii] ("The possibility of multiple suits against an infringer by assignees of related patents has long been recognized as one of the concerns behind the doctrine of double patenting.").

See In re Hubbell, 709 F.3d 1140, 1145 (Fed. Cir. 2013).

Analyzing obviousness-type double patenting takes a two-step process. First, the differences between the claims of the patent being used as a reference and the claims at issue are identified. *Georgia-Pacific v. US Gypsum*, 195 F.3d 1322, 1326 (Fed. Cir. 1999) ("[A]nalysis of the claims at issue is the first step in determining if the second invention is merely an obvious variation of the first.... Because these two claims are so similar, we must look to see if there is anything to distinguish claim 1 of the '989 patent from claim 1 of the '569 patent. There are two differences between these claims."), *amended on rehearing*, 204 F.3d 1359 (Fed. Cir. 2000).

Second, whether those differences raise a patentable distinction must be determined. *Georgia-Pacific*, 195 F.3d at 1328 (finding that "[t]hese differences are not sufficient to render the claims patentably distinct"). "A later patent claim 'is not patentably distinct from an earlier claim if the later claim is obvious over, or anticipated by, the earlier claim.'" *Hubbell*, 709 F.3d at 1145 (*quoting Eli Lilly & Co. v. Barr Labs., Inc.*, 251 F.3d 955, 968 (Fed. Cir. 2001)). Where the recited claims at issue are not explicitly disclosed in the claims of the patent being used as a reference, Samsung may rely on the knowledge of a person of ordinary skill in the art, admitted prior art, or disclosure in the prior art references relied upon by Samsung.

A. ’544 patent

Claim 1 of the ’544 patent is invalid due to obviousness-type double patenting in view of the claimed invention in at least the separately-asserted ’433 patent. The ’544 and ’433 patents share common inventors and are related for double patenting purposes. The ’433 patent expires before the ’544 patent, and therefore qualifies as a double patenting reference. The ’433 patent expires on March 2, 2029, while the ’544 patent expires on February 3, 2030. *See Gilead Scis., Inc. v. Natco Pharma Ltd.*, (holding that an earlier expiring patent would “qualify as an obviousness-type double patenting reference for a later-expiring patent”); *Abbvie v. Mathilda & Terence Kennedy Institute*, 764 F.3d 1355, 1373 (finding the crucial purpose of the obviousness-type double patenting doctrine “to prevent an inventor from securing a second, later expiring patent for the same invention” still exists, due, for example, to patents that have “different patent terms due to examination delays at the PTO”).

A table showing the overlap in claim language between claim 1 of the ’544 patent and claims 1, 2, and 13 of the ’433 patent is provided below:

’544 Patent Claim 1	’433 Patent Claim 1
A wireless end-user device, comprising:	A wireless end-user device, comprising:
a wireless modem to communicate data for network service usage activities between the device and a wireless network;	a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the at least one WWAN, the at least one WWAN having a corresponding network type of a plurality of wireless network types supported by the device for data communication;
	a non-transitory memory to store a network service activity control policy set, the policy set including at least a first differential traffic control policy element associating one or more Internet activity access controls with at least a first end-user application; and
	one or more processors configured to

a processor that executes instructions to associate network service usage activity, on behalf of a first device application, and that occurs when the first device application is not in the foreground of user interaction, with a network service usage control policy,	access the network service activity control policy set, determine whether to apply the one or more Internet activity access controls with respect to a first Internet access request by or on behalf of the first end-user application, based at least on which of the plurality of wireless network types is to provide data communication for Internet access requests, and when the one or more Internet activity access controls are to be applied, apply the one or more Internet activity access controls to aggregate network activity for the first Internet access request with network activity for one or more other data communication requests, which are otherwise not associated with the first end-user application, before allowing network activity in association with the first Internet access request.
	'433 Patent Claim 2
	2. The wireless end-user device of claim 1, wherein the first end-user application is capable of both interacting with a user in a user interface foreground of the device, and at least some Internet service activities when not interacting with a user in the device user interface foreground, the one or more processors further configured to
set an application state indicating whether the first device application, associated with a particular network service usage activity, is in the foreground of user interaction, and	classify whether the first end-user application is interacting with the user in the device user interface foreground, and
dynamically determine whether to apply the network service usage control policy to the particular network service usage activity, based on the application state and based on a power control state; and	determine whether to apply the one or more Internet activity access controls based also on the classification of whether the first end-user application is interacting with the user.
	'433 Patent Claim 13
a memory coupled to the processor to provide the processor with the instructions.	The wireless end-user device of claim 1, wherein the one or more processors are further configured to dynamically change the determination of whether to apply the one or

	more Internet activity access controls based on a power state of the device.

While the claims of the ’433 patent contain minor linguistic differences from the independent claim of the ’544 patent, none of those differences renders the independent claim of the ’544 patent as a whole patentably distinct from the claims in the ’433 patent. Like claim 1 of the ’544 patent, claims 1, 2 and 13 of the ’433 patent—taken together—teach a “wireless end-user device, comprising”: (1) a wireless modem to communicate data for Internet (i.e., network) service activities between the device and a wireless network (claim 1 of the ’433 patent); (2) a processor configured to access the network service activity policy set and associate the network service activity with a first application (claim 1 of the ’433 patent) when the application is not in the foreground of user interaction (claim 2 of the ’433 patent), set an application state by classifying whether the first application is in the foreground of user interaction (claim 2 of the ’433 patent), and determine whether to apply the control policy based on the application state (claim 2 of the ’433 patent) and based on a power control state (claim 13 of the ’433 patent); and (3) a memory (claim 1 of the ’433 patent).

It would have been obvious to a POSITA that the “wireless wide area network (WWAN) modem” of claim 1 of the ’433 patent is a “wireless modem” as required by claim 1 of the ’544 patent. A POSITA would have also found “communicat[ing] data for network service usage activities between the device and a wireless network” (’544 patent claim 1) obvious in view of claim 1 of the ’433 patent’s requirement for “communicat[ing] data for Internet service activities between the device and at least one WWAN” (’433 patent claim 1). A POSITA would have also found “execut[ing] instructions to associate network service usage activity, on behalf of a first device application … with a network service usage control policy” (’544 patent claim 1) obvious in view of claim 1 of the ’433 patent, which requires “access[ing] the network service activity control policy set," and "determin[ing] whether to apply the one or more Internet activity access controls with respect to a first Internet access request by or on behalf of the first end-user application" ('433 patent claim 1). A POSITA would have found "execut[ing] instructions to associate network service usage activity, on behalf of a first device application… when the first device application is not in the foreground of user interaction" ('544 patent claim 1) obvious in view of claim 2 of the '433 patent, which requires "at least some Internet service activities when not interacting with a user in the device user interface foreground." A POSITA would have also found "set[ting] an application state indicating whether the first device application, associated with a particular network service usage activity, is in the foreground of user interaction" ('544 patent claim 1) obvious in view of claim 2 of the '433 patent, which requires "classify[ing] whether the first end-user application is interacting with the user in the device user interface foreground." And finally, a POSITA would have found "determin[ing] whether to apply the network service usage control policy to the particular network service usage activity, based on the application state and based on a power control state" ('544 patent claim 1) obvious in view of claims 2 and 13 of the '433 patent, which require "determin[ing] whether to apply the one or more Internet activity access controls based also on the classification of whether the first end-user application is interacting with the user [i.e., in the foreground]" and "dynamically chang[ing] the determination of whether to apply the one or more Internet activity access controls based on a power state of the device," respectively.

Another table showing the overlap in claim language between claims 2, 3, 7, 11, 17, and 23 of the '544 patent and claims 3, 4, 13, 10, 16, and 18 of the '433 patent, respectively, is provided below:

'544 Patent Claim 2	**'433 Patent Claim 3**

The wireless end-user device of claim 1, wherein the processor executing the instructions determines that the first device application is in the foreground of user interaction when the user of the device is directly interacting with that application or perceiving any benefit from that application.	The wireless end-user device of claim 2, wherein the one or more processors are configured to classify that the first end-user application is interacting with the user in the device user interface foreground when the user of the device is directly interacting with that application or perceiving any benefit from that application.
'544 Patent Claim 3	**'433 Patent Claim 4**
The wireless end-user device of claim 1, wherein the processor executing the instructions determines that the first device application is in the foreground of user interaction based on a state of user interface priority for the application.	The wireless end-user device of claim 2, wherein the one or more processors are configured to classify that the first end-user application is interacting with the user in the device user interface foreground based on a state of user interface priority for the application.
'544 Patent Claim 7	**'433 Patent Claim 13**
The wireless end-user device of claim 1, wherein the power control state is a power state of the device.	The wireless end-user device of claim 1, wherein the one or more processors are further configured to dynamically change the determination of whether to apply the one or more Internet activity access controls based on a power state of the device.
'544 Patent Claim 11	**'433 Patent Claim 10**
The wireless end-user device recited in claim 1, wherein dynamically determining whether to apply the network service usage control policy to the particular network service usage activity is further based on a current wireless network and/or the network service usage control policy is based on a current wireless network.	The wireless end-user device of claim 1, wherein the one or more processors are further configured to select the first differential traffic control policy element from the network service activity control policy set based at least in part on which of the plurality of wireless network types is to provide data communication for Internet access requests.
'544 Patent Claim 17	**'433 Patent Claim 16**
The wireless end-user device recited in claim 1, wherein the network service usage control policy includes traffic control policy filters using a network busy state and/or a time of day as an index into a traffic control setting.	The wireless end-user device of claim 1, wherein the one or more Internet activity access controls further limit the allowed network activity to particular time windows.
'544 Patent Claim 23	**'433 Patent Claim 18**
The wireless end-user device of claim 1, wherein the power control state is a power state of the wireless modem.	The wireless end-user device of claim 1, wherein the one or more Internet activity access controls further prevent the first Internet access request from causing a change to a power state of the modem.

While the dependent claims of the ’433 patent contain minor linguistic differences from the dependent claims of the ’544 patent, none of those differences renders the above dependent claims of the ’544 patent as a whole patentably distinct from the claims in the ’433 patent. For example, a POSITA would have found “determin[ing] that the first device application is in the foreground of user interaction when the user of the device is directly interacting with that application or perceiving any benefit from that application” (’544 patent claim 2) obvious in view of claim 3 of the ’433 patent, which requires “classify[ing] that the first end-user application is interacting with the user in the device user interface foreground when the user of the device is directly interacting with that application or perceiving any benefit from that application.” A POSITA would have further found a “processor executing the instructions determines that the first device application is in the foreground of user interaction based on a state of user interface priority for the application” (’544 patent claim 3) obvious in view of claim 4 of the ’433 patent, which requires a processor “configured to classify that the first end-user application is interacting with the user in the device user interface foreground based on a state of user interface priority for the application.” A POSITA would have also found “wherein the power control state is a power state of the device” (’544 patent claim 7) obvious in view of claim 13 of the ’433 patent, which also discusses using the “power state of the device.” A POSITA would have found “determining whether to apply the network service usage control policy to the particular network service usage activity is further based on a current wireless network and/or the network service usage control policy is based on a current wireless network” (’544 patent claim 11) obvious in view of claim 10 of the ’433 patent, which requires “select[ing] the first differential traffic control policy element from the network service activity control policy set based at least in part on which of the plurality of wireless network types is to provide data communication for Internet access requests.” Indeed, both claim 11 of the ’544 patent and claim 10 of the ’433 patent cover deciding whether to apply a network service usage control policy / differential traffic control policy from the network service activity control policy set based on the wireless network. A POSITA would have also found “the network service usage control policy includes traffic control policy filters using … a time of day” (’544 patent claim 17) obvious in view of claim 16 of the ’433 patent, which requires using access controls to limit network activity to a particular time window (i.e., time of day). Finally, a POSITA would have found “wherein the power control state is a power state of the wireless modem” (544 patent claim 23) obvious in view of claim 18 of the ’433 patent which similarly discusses the “power state of the modem.”

B. ’578 patent

Claim 1 of the ’578 patent is invalid due to obviousness-type double patenting in view of the claimed invention in at least the separately-asserted ’976 patent. The ’578 and ’976 patents share common inventors and are related for double patenting purposes. The ’976 patent expires before the ’578 patent, and therefore qualifies as a double patenting reference. The ’976 patent expires on March 2, 2029, while the ’578 patent expires on April 22, 2029. *See Gilead Scis., Inc. v. Natco Pharma Ltd.*, (holding that an earlier expiring patent would “qualify as an obviousness-type double patenting reference for a later-expiring patent”); *Abbvie v. Mathilda & Terence Kennedy Institute*, 764 F.3d 1355, 1373 (finding the crucial purpose of the obviousness-type double patenting doctrine “to prevent an inventor from securing a second, later expiring patent for the same invention” still exists, due, for example, to patents that have “different patent terms due to examination delays at the PTO”).

A table showing the overlap in claim language between claim 1 of the ’578 patent and claims 1 and 4 of the ’976 patent is provided below:

’578 Patent Claim 1	’976 Patent Claim 1
A wireless end-user device, comprising:	A wireless end-user device, comprising:
a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the at least one WWAN;	a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the WWAN;
a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the at least one WLAN;	a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN;
a non-transitory memory to store a differential traffic control policy applicable to data communicated for Internet service activities using the WWAN modem and the at least one WWAN, but not applicable to data communicated for Internet service activities using the WLAN modem and the at least one WLAN;	
a user interface to allow a user to set one or more of a plurality of aspects of the differential traffic control policy to select one or more applications that are only allowed to utilize the at least one WWAN for Internet service activities when those applications are classified as interacting with a user in the device user interface foreground; and	
one or more processors configured to implement an application program interface (API) that allows a particular application to access one or more aspects of the differential traffic control policy applicable to that application, including whether the user-settable aspects of the policy only allow the particular application to utilize the at least one WWAN for Internet service activities when the particular application is classified as interacting with a user in the device user interface foreground.	one or more processors configured to determine, for a first end-user application capable of running in a background state and capable of running as a foreground application, whether the application is running in a background state or as a foreground application, and control, via an application program interface (API), application access for Internet service activities provided through the WWAN modem and the WLAN modem, to, based on a first differential traffic control policy, selectively block and allow access by the first end-user application to the WWAN modem at a time when data for Internet service activities is communicated through a WWAN modem connection to the at least one WWAN,

	wherein the access is selectively blocked based on a determination that the first end-user application is running in a background state, and wherein the access is selectively allowed based on a determination that the first end-user application is running as a foreground application.
	’976 Patent Claim 4
	4. The wireless end-user device of claim 1, further comprising a user interface to inform the user of the device when there are options to set, control, override, or modify service usage controls that affect the first differential traffic control policy.

While the claims of the ’976 patent contain minor linguistic differences from the independent claim of the ’578 patent, none of those differences renders the independent claim of the ’578 patent as a whole patentably distinct from the claims in the ’976 patent. For example, a POSITA would have found the “wireless end-user device,” comprising “a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the at least one WWAN” and “a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the at least one WLAN” (’578 patent claim 1) obvious in view of claim 1 of the ’976 patent, which similarly recites a “wireless end-user device,” comprising “a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the WWAN” and “a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN.”

A POSITA would have also found “one or more processors configured to implement an application program interface (API) that allows a particular application to access one or more aspects of the differential traffic control policy applicable to that application, including whether the user-settable aspects of the policy only allow the particular application to utilize the at least one WWAN for Internet service activities when the particular application is classified as interacting with a user in the device user interface foreground" ('578 patent claim 1) obvious in view of claim 1 of the '976 patent, which similarly requires one or more processors configured to "control, via an application program interface (API), application access for Internet service activities provided through the WWAN modem" and, "based on a first differential traffic control policy, selectively block and allow access by the first end-user application to the WWAN modem," "wherein the access is selectively blocked based on a determination that the first end-user application is running in a background state, and wherein the access is selectively allowed based on a determination that the first end-user application is running as a foreground application." In other words, both claims require implementing an API that allows a particular application to access one or more aspects of the differential traffic control policy, including only allowing the particular application to utilize the WWAN for Internet service activities when the application is classified as interacting with a user in the device user interface foreground rather than running in a background state. A POSITA would have also found "a user interface to allow a user to set one or more of a plurality of aspects of the differential traffic control policy" ('578 patent claim 1) obvious in view of claim 4 of the '976 patent, which also recites "a user interface to inform the user of the device when there are options to set, control, override, or modify service usage controls that affect the first differential traffic control policy."

Finally, despite similar language missing from the claims of the '976 patent, a POSITA would have found "a non-transitory memory to store a differential traffic control policy applicable to data communicated for Internet service activities using the WWAN modem and the at least one

WWAN, but not applicable to data communicated for Internet service activities using the WLAN modem and the at least one WLAN" ('578 patent claim 1) obvious in view of claims 1 and 4 of the '976 patent. Indeed, a POSITA would have readily understood that the applicable "differential traffic control policy" recited in claim 1 of the '976 patent would be, or could be, stored in a non-transitory memory located within the claimed "wireless end-user device" and that it would be obvious to do so.

Another table showing the overlap in claim language between claims 3, 4, 5, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 21, and 22 of the '578 patent and claims 1, 2, 14, 3, 5, 6, 7, 11, 9, 10, 25, 26, 27, 28, 29, 16, 18, and 20 of the '976 patent, respectively, is provided below:

'578 Patent Claim 3	'976 Patent Claim 1
The wireless end-user device of claim 1, the API further to indicate, to the particular application, one or more network access conditions based on the differential traffic control policy, wherein the one or network access conditions include a network access condition that indicates the unavailability to the particular application of an Internet data service that is currently available via the WWAN modem to a different application.	indicate to the first end-user application, via an application program interface (API), one or more network access conditions based on the applied first differential traffic control policy, including a first network access condition that indicates the unavailability to the first end-user application, when the first end-user application is classified as not interacting in the device display foreground with the user, of Internet data service that is available via the WWAN modem
'578 Patent Claim 4	**'976 Patent Claim 2**
The wireless end-user device of claim 1, wherein the one or more processors are further configured to classify that the particular application is interacting with the user in the device user interface foreground when a user of the device is directly interacting with the particular application or perceiving any benefit from the particular application.	The wireless end-user device of claim 1, wherein the one or more processors are configured to classify that the first end-user application is not interacting in the device display foreground with the user when the user of the device is not directly interacting with that application or perceiving any benefit from that application.
'578 Patent Claim 5	**'976 Patent Claim 14**
The wireless end-user device of claim 1, wherein the one or more processors are further configured to classify that the particular application is interacting with the user in the device user interface foreground based on a	The wireless end-user device of claim 1, wherein the one or more processors configured to classify whether or not the first end-user application, when running, in interacting in the device display foreground with a user perform the classification based at least in part on a

state of user interface priority for the application.	state of user interface priority for the application.
’578 Patent Claim 7	**’976 Patent Claim 3**
The wireless end-user device of claim 1, the user interface to provide a user of the device with information regarding why the differential traffic control policy is applied to the particular application.	The wireless end-user device of claim 1, further comprising a user interface to provide the user of the device with information regarding why the first differential traffic control policy is applied to the first end-user application.
’578 Patent Claim 8	**’976 Patent Claim 5**
The wireless end-user device of claim 1, wherein the differential traffic control policy is part of a multimode profile having different policies for different networks.	The wireless end-user device of claim 1, wherein the first differential traffic control policy is part of a multimode profile having different policies for different networks.
’578 Patent Claim 9	**’976 Patent Claim 6**
The wireless end-user device of claim 8, wherein the one or more processors are further configured to select a traffic control policy from the multimode profile based at least in part on the type of network connection currently in use by the device.	The wireless end-user device of claim 5, wherein the one or more processors are further configured to select a traffic control policy from the multimode profile based at least in part on the type of network connection currently in use by the device.
’578 Patent Claim 10	**’976 Patent Claim 7**
The wireless end-user device of claim 9, wherein the one or more processors are further configured to, when the type of network connection is at least one type of WLAN connection, select a traffic control policy from the multimode profile based at least in part on a type of network connection from the WLAN to the Internet.	The wireless end-user device of claim 6, wherein the one or more processors are further configured to, when the type of network connection is at least one type of WLAN connection, select a traffic control policy from the multimode profile based at least in part on a type of network connection from the WLAN to the Internet.
’578 Patent Claim 11	**’976 Patent Claim 11**
The wireless end-user device of claim 8, wherein the differential traffic control policy is the policy for a roaming WWAN network, the multimode profile having a second traffic control policy for a home WWAN network.	The wireless end-user device of claim 1, wherein the one or more processors apply the first differential traffic control policy to one of but not both of a connection to a roaming WWAN network and a connection to a home WWAN network.
’578 Patent Claim 12	**’976 Patent Claim 9**
The wireless end-user device of claim 1, the one or more processors further comprising a network stack interface in communication with the API.	The wireless end-user device of claim 1, further comprising a network stack interface integrated with the API.
’578 Patent Claim 13	**’976 Patent Claim 10**
The wireless end-user device of claim 1, further comprising a networking stack, wherein the one or more processors are further	The wireless end-user device of claim 1, further comprising a networking stack, wherein the one or more processors are further

configured to, at an application service interface layer, identify application traffic flows prior to the flows entering the networking stack.	configured to, at an application service interface layer, identify application traffic flows prior to the flows entering the networking stack.
’578 Patent Claim 14	**’976 Patent Claim 25**
The wireless end-user device of claim 1, wherein the API comprises a network access API.	The wireless end-user device of claim 1, wherein the API comprises a network access API.
’578 Patent Claim 15	**’976 Patent Claim 26**
The wireless end-user device of claim 1, wherein the API further allows the particular application to access information indicating whether a current connected WWAN is a roaming network or a non-roaming network.	The wireless end-user device of claim 1, wherein the one or more network access conditions indicated via the API to the first end-user application comprises information on whether a current connected WWAN is a roaming network or a non-roaming network.
’578 Patent Claim 16	**’976 Patent Claim 27**
The wireless end-user device of claim 1, wherein the API further informs the particular application when it is allowed to access Internet data service that is currently available via the WWAN modem.	The wireless end-user device of claim 1, wherein the API informs the first end-user application when it is allowed to access Internet data service that is available via the WWAN modem.
’578 Patent Claim 17	**’976 Patent Claim 28**
The wireless end-user device of claim 1, wherein the API informs the particular application of one or more network traffic controls that the application is expected to implement.	The wireless end-user device of claim 1, wherein the API informs the first end-user application of one or more network traffic controls that the first end-user application is expected to implement.
’578 Patent Claim 18	**’976 Patent Claim 29**
The wireless end-user device of claim 1, wherein the API instructs the particular application to transition to a different state.	The wireless end-user device of claim 1, wherein the API instructs the first end-user particular application to transition to a different state.
’578 Patent Claim 19	**’976 Patent Claim 16**
The wireless end-user device of claim 1, wherein the one or more processors are configured to associate the particular application with the differential traffic control policy based on an application behavior.	The wireless end-user device of claim 1, wherein the one or more processors are configured to associate the first end-user application with the first differential traffic control policy based on an application behavior.
’578 Patent Claim 21	**’976 Patent Claim 18**
The wireless end-user device of claim 1, wherein the one or more processors are further configured to update the differential traffic control policy based on information received from a network element.	The wireless end-user device of claim 1, wherein the one or more processors are further configured to update the first differential traffic control policy based on information received from a network element.
’578 Patent Claim 22	**’976 Patent Claim 20**

The wireless end-user device of claim 1, wherein the one or more processors are configured to apply the differential traffic control policy to selectively block network access by the particular application by intercepting open, connect, and/or write requests by the particular application to a network stack.	The wireless end-user device of claim 1, wherein the one or more processors configured to apply the first differential traffic control policy to disallow Internet service activity on behalf of the first end-user application perform a disallowance of Internet service activity by intercepting open, connect, and/or write requests by the first end-user application to a network stack.

While the claims of the ’976 patent contain minor linguistic differences from the dependent claims of the ’578 patent, the language is largely identical and none of those differences renders the above dependent claims of the ’578 patent as a whole patentably distinct from the claims in the ’976 patent.

VI. INVALIDITY UNDER 35 U.S.C. § 112

Headwater has not yet provided a claim construction for many of the terms and phrases that Samsung anticipates will be in dispute. Samsung, therefore, cannot provide a complete list of its § 112 defenses because Samsung does not know whether Headwater will proffer a construction for certain terms and phrases that is broader than, or inconsistent with, the construction that would be supportable by the disclosure set forth in the specification.

Nevertheless, Samsung contends that, at least under Headwater’s actual and/or apparent application of the claims, the Asserted Claims are invalid based on inadequate written description and/or a lack of enablement under 35 U.S.C. § 112 ¶ 1, and/or based on indefiniteness under 35 U.S.C. § 112 ¶ 2.

Samsung’s aforementioned identification of prior art that anticipates and/or renders obvious particular claim elements, including the attached claim charts, should not be deemed as an admission that any claim element satisfies the requirements of 35 U.S.C. § 112. While Samsung asserts below that a claim is invalid under 35 U.S.C. § 112 (such as because of a failure to particularly point out and distinctly claim the alleged invention, failure to provide written description support in the specification, and/or failure to enable one of ordinary skill in the art to make and use the alleged invention), Samsung has nonetheless provided prior art disclosures that anticipate or render obvious the claim on the assumption that Headwater will contend those claims are definite, are supported by an adequate written description, and are adequately enabled.

A. Lack of Written Description and Enablement Under 35 U.S.C. § 112 ¶ 1

Certain claims in the Asserted Patents are invalid for lack of written description. Section 112 requires that a patent specification "contain a written description . . . of the manner and process of making and using [the invention] in such full, clear, concise and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same." 35 U.S.C. § 112 ¶ 1. A patent's written description "must clearly allow persons of ordinary skill in the art to recognize that [the inventor] invented what is claimed." *Ariad Pharms., Inc. v. Eli Lilly & Co.*, 598 F.3d 1336, 1351 (Fed. Cir. 2010) (en banc). The disclosure must "convey to those skilled in the art that the inventor had possession of the claimed subject matter as of the filing date." *Id.* The level of detail required to satisfy the written description requirement varies depending on the nature and scope of the claims and on the complexity and predictability of the relevant technology, but a "mere wish or plan" for obtaining the alleged invention does not satisfy the written description requirement. *Novozymes A/S v. DuPont Nutrition Biosciences APS*, 723 F.3d 1336, 1344 (Fed. Cir. 2013). Put another way, "a description that merely renders the invention obvious does not satisfy the requirement." *Ariad*, 598 F.3d at 1351. Instead, "all the limitations must appear in the specification." *Lockwood v. Am. Airlines, Inc.*, 107 F.3d 1565, 1572 (Fed. Cir. 1997). Samsung contends that, at least under Headwater's actual and/or apparent application of the claims, the specifications of at least one or more of the Asserted Patents do not include a sufficient written description supporting the claims. Moreover, Samsung contends that Headwater's actual and/or apparent application of the Asserted Claims covers a broader scope than is justified and/or supported by the written description provided in the specifications of at least one or more of the Asserted Patents. *Tronzo v. Biomet, Inc.*, 156 F.3d 1154, 1159 (Fed. Cir. 1998); *LizardTech, Inc. v. Earth Res. Mapping, Inc.*, 424 F.3d 1336, 1346 (Fed. Cir. 2005); *ICU Med., Inc. v. Alaris Med. Sys., Inc.*, 558 F.3d 1368 (Fed. Cir. 2009).

Section 112 likewise requires that the specification "enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the" alleged invention. 35 U.S.C. § 112 ¶ 1. A claim is not enabled if, "at the effective filing date of the patent, one of ordinary skill in the art could not practice their full scope without undue experimentation." *Wyeth and Cordis Corp. v. Abbott Labs.*, 720 F.3d 1380, 1384 (Fed. Cir. 2013). "This important doctrine prevents both inadequate disclosure of an invention and overbroad claiming that might otherwise attempt to cover more than was actually invented." *MagSil Corp. v. Hitachi Global Storage Techs., Inc.*, 687 F.3d 1377, 1381 (Fed. Cir. 2012). Samsung contends that, at least under Headwater's actual and/or apparent application of the claims, the specifications of at least one or more of the Asserted Patents do not enable any person skilled in the relevant art to make and use the alleged inventions of the Asserted Claims without undue experimentation.

Furthermore, under Headwater's actual and/or apparent application of the claims, the specifications of at least one or more of the Asserted Patents do not enable the broad scope of the Asserted Claims as Headwater asserts. Samsung contends that Headwater's actual and/or apparent application of the Asserted Claims covers a broader scope than is justified, and certainly broader than is enabled in the specifications. As explained below, the specifications of at least one or more of the Asserted Patents have not enabled a person of ordinary skill in the art at the time of the alleged invention to perform the full scope of all Asserted Claims.

Each of the asserted claims below are invalid because, at least to the extent Headwater contends any of the following limitations should be construed to encompass Samsung's accused instrumentalities, the specifications fail to provide written description and/or an enabling disclosure of at least the following limitations:

1. '701 patent

- **Claims 1, 3, 4, 5, 8, 11-19, 23, 24:** "differential traffic control policy"
- **Claim 1:** "selectively block and allow access by the first end-user application to the WWAN modem at a time when data for Internet service activities is communicated through a WWAN modem connection"
- **Claim 1:** "the access is selectively blocked based on a determination that the first end-user application is running in a background state, and wherein the access is selectively allowed based on a determination that the first end-user application is running as a foreground application"
- **Claim 3:** "provide a user of the device with information regarding why the first differential traffic control policy is applied"
- **Claims 5-7:** "multimode profile"
- **Claim 8:** "second differential traffic control policy that can be set different from the first differential traffic control policy"

2. '184 patent

- **Claim 1:** "classify whether a particular application associated with an Internet service access request, and capable of both interacting with a user in a user interface foreground of the device, and at least some Internet service activities when not interacting with a user in the device user interface foreground, is interacting with the user in the device user interface foreground"

- **Claim 1, 6, 7, 9, 10, 11, 12, 13:** "differential traffic control policy"
- **Claim 1, 6, 13:** "a differential traffic control policy list"
- **Claim 1:** "based on an application-specific amount of Internet data usage reaching a limit"
- **Claim 2:** "perceiving any benefit from that application"
- **Claim 3:** "classify that the particular application is interacting with the user in the device user interface foreground based on a state of user interface priority for the application"
- **Claim 5:** "delay network activity in association with the first Internet access request until a second Internet access request is received and allowed"
- **Claim 6:** "wherein the user interface is to inform the user of the device when there are options to set, control, override, or modify at least one aspect of the differential traffic control policy and/or the differential traffic control policy list"
- **Claim 8:** "for a time when the WLAN modem is to provide data communication for Internet service activities, not count an application-specific amount of Internet data usage toward the limit"
- **Claim 10:** "dynamically change a determination of whether to apply the differential traffic control policy based on a power state of the device"

3. '578 patent

- **Claims 1-3, 7, 8, 11, 19-22:** "differential traffic control policy"
- **Claim 1:** "user interface to allow a user to set one or more of a plurality of aspects of the differential traffic control policy"

- **Claim 1:** "applications are classified as interacting with a user in the device user interface foreground"
- **Claim 1:** "implement an application program interface (API) that allows a particular application to access one or more aspects of the differential traffic control policy applicable to that application."
- **Claim 4:** "perceiving any benefit from the particular application"
- **Claim 5:** "based on a state of user interface priority for the application"
- **Claims 8-11:** "multimode profile"

4. '445 patent

- **Claim 1:** "classify, as a first classification, whether data for Internet service activities is to be communicated through the WWAN modem or the WLAN modem"
- **Claim 1:** "classify, as a second classification, whether a particular application associated with an Internet service access request . . . is interacting with the user in the device user interface foreground."
- **Claims 1, 3-5, 8-15:** "differential traffic control policy"
- **Claim 1:** "block the Internet service access request in a first state of the first and second classifications"
- **Claim 2:** "perceiving any benefit from the particular application"
- **Claims 5-7:** "multimode profile"
- **Claim 8:** "one or more processors are operable to allow the Internet service access request when data for Internet service activities is classified as to be provided through the WWAN modem, and the particular application is not classified as interacting with a user in the device user interface foreground"

- **Claim 16:** "selectively block the Internet service access request by intercepting open, connect, and/or write requests"
- **Claim 19:** "classify that the particular application is interacting with a user in the device user interface foreground based on a state of user interface priority for the application"
- **Claim 20:** "second one or more applications are not subject to a differential network access control that is applicable to the first one or more applications"

5. '224 patent

- **Claim 1:** "for each given application of a plurality of applications on the wireless end-user device"
- **Claim 1:** "associate each given one of the network service usage activities with a service usage control policy dynamically selected from the set of service usage control policies"
- **Claim 1:** "wherein the differential network access control comprises a set of service usage control policies applicable when a network service is available via the at least one wireless modem"
- **Claim 1:** "a first policy that allows the given network service usage activity to currently communicate data with a network destination via the at least one wireless modem"
- **Claim 1:** "a second policy that defers data communication associated with the given network service usage activity until a device state change occurs"
- **Claim 3:** "wherein the dynamically selected service usage control policy for a given network service usage activity is further selected based on whether the application associated with that network service usage activity is currently in the foreground of user interaction"

- **Claim 4:** "wherein the set of service usage control policies further comprises a third policy that blocks data communication associated with the given network service usage activity"
- **Claim 7:** "modem power save state"
- **Claim 10:** "wherein for the second policy that defers data communication associated with the given network service usage activity until the device state change occurs, the device state change comprises a change in user interaction with the wireless end-user communications device"
- **Claim 13:** "wherein the processor is further configured to monitor network service usage behavior for each of the plurality of applications, and wherein the dynamic selection of the service usage control policy for a given network service usage activity is further based on a comparison of the monitored network service usage behavior for the application associated with the given network service usage activity to expected access limits for that application"

6. '976 patent

- **Claim 1:** "classify, for a first end-user application capable of interacting in the device display foreground with a user and capable of at least some Internet service activity when not interacting in the device display foreground with the user, whether or not the first end-user application, when running, is interacting in the device display foreground with the user"
- **Claim 1:** "for a time period when data for Internet service activities is communicated through a WWAN modem connection to the at least one WWAN, apply a first differential traffic control policy to Internet service activity on behalf of the first end-user application, such that Internet service activity on behalf of the first end-user application is disallowed

when the one or more processors classify the first end-user application as not interacting in the device display foreground with the user"

- **Claim 1:** "indicate to the first end-user application, via an application program interface (API), one or more network access conditions based on the applied first differential traffic control policy"
- **Claim 1:** "a first network access condition that indicates the unavailability to the first end-user application, when the first end-user application is classified as not interacting in the device display foreground with the user, of Internet data service that is available via the WWAN modem"
- **Claim 1:** "a second network access condition that indicates the availability to the first end-user application, when the first end-user application is classified as interacting in the device display foreground with the user, of Internet data service that is available via the WWAN modem"
- **Claim 2:** "wherein the one or more processors are configured to classify that the first end-user application is not interacting in the device display foreground with the user when the user of the device is not directly interacting with that application or perceiving any benefit from that application"
- **Claims 1, 3-5, 8, 11-13, 15-20, 23, 24:** "differential traffic control policy"
- **Claims 5-7:** "multimode profile"
- **Claim 7:** "wherein the one or more processors are further configured to, when the type of network connection is at least one type of WLAN connection, select a traffic control policy from the multimode profile based at least in part on a type of network connection from the WLAN to the Internet"

- **Claim 12:** "wherein the one or more processors are further configured to dynamically change the application of the first differential traffic control policy based on a power state of the device"
- **Claim 13:** "wherein the one or more processors are further configured to dynamically change the application of the first differential traffic control policy based on a device usage state"

7. '433 patent

- **Claim 1:** "determine whether to apply the one or more Internet activity access controls with respect to a first Internet access request by or on behalf of the first end-user application, based at least on which of the plurality of wireless network types is to provide data communication for Internet access requests"
- **Claim 1:** "when the one or more Internet activity access controls are to be applied, apply the one or more Internet activity access controls to aggregate network activity for the first Internet access request with network activity for one or more other data communication requests, which are otherwise not associated with the first end-user application, before allowing network activity in association with the first Internet access request"
- **Claims 1, 7, 10:** "differential traffic control policy"
- **Claims 1, 10, 17:** "network service activity control policy set"
- **Claim 2:** "classify whether the first end-user application is interacting with the user in the device user interface foreground, and determine whether to apply the one or more Internet activity access controls based also on the classification of whether the first end-user application is interacting with the user"

- **Claim 3:** "wherein the one or more processors are configured to classify that the first end-user application is interacting with the user in the device user interface foreground when the user of the device is directly interacting with that application or perceiving any benefit from that application"
- **Claim 4:** "wherein the one or more processors are configured to classify that the first end-user application is interacting with the user in the device user interface foreground based on a state of user interface priority for the application"
- **Claim 5:** "wherein the one or more processors are configured to classify that the first end-user application is not interacting with the user in the device user interface foreground when the application is providing or utilizing a background data service"
- **Claims 7-8:** "application list"
- **Claim 13:** "wherein the one or more processors are further configured to dynamically change the determination of whether to apply the one or more Internet activity access controls based on a power state of the device"
- **Claim 14:** "wherein the one or more processors are further configured to dynamically change the determination of whether to apply the one or more Internet activity access controls based on a device usage state"

8. '544 patent

- **Claim 1**: "a processor that executes instructions to associate network service usage activity, on behalf of a first device application, and that occurs when the first device application is not in the foreground of user interaction, with a network service usage control policy"
- **Claim 1:** "set an application state indicating whether the first device application, associated with a particular network service usage activity, is in the foreground of user interaction"

- **Claim 1:** "dynamically determine whether to apply the network service usage control policy to the particular network service usage activity, based on the application state and based on a power control state"
- **Claim 2:** "wherein the processor executing the instructions determines that the first device application is in the foreground of user interaction when the user of the device is directly interacting with that application or perceiving any benefit from that application"
- **Claim 3:** "wherein the processor executing the instructions determines that the first device application is in the foreground of user interaction based on a state of user interface priority for the application"
- **Claim 7:** "wherein the power control state is a power state of the device"
- **Claim 22:** "wherein the power control state is a power save state of the device"
- **Claims 1, 4, 5, 6, 11, 12, 18-21:** "network service usage activity"
- **Claims 1, 4, 5, 11-19:** "network service usage control policy"

9. ’773 patent

- **Claim 1:** "wherein at least one such service usage control policy specifies that during a time when the current WWAN network busy state indicates network congestion, a selected subset of the background network service usage activities are deferred until network congestion is no longer indicated"
- **Claim 1:** "allow each of the background network service usage activities to access the WWAN during the respective deferred time slot for that background network service usage activity"
- **Claim 1, 2, 16, 17, 18:** "service usage control policy"

- **Claim 9:** "wherein the instructions provided to the processor further cause the processor to select different service usage control policies based at least in part on, in addition to the current WWAN network busy state, monitoring of user interaction with the device"
- **Claim 12:** "wherein to receive respective requests from a plurality of applications comprises receiving the requests through an operating system device service access Application Programming Interface (API), the API allowing applications to indicate a data size for a background network service usage activity"
- **Claim 16:** "service usage behavior analysis"
- **Claim 18:** "wherein the instructions provided to the processor further cause the processor to apply the current service usage control policy for one or more of the background network service activities based on monitoring user interaction with the activity"

Samsung's investigation of grounds of invalidity based upon failure to meet the written description and/or enablement requirements is ongoing, and Samsung reserves the right to supplement these Contentions.

B. Indefiniteness Under 35 U.S.C. § 112 ¶ 2

35 U.S.C. § 112, ¶ 2 requires that a patent claim "particularly point[] out and distinctly claim[] the subject matter which the applicant regards as his invention." 35 U.S.C. § 112, ¶ 2. Claim terms that fail to inform those skilled in the art "with reasonable certainty . . . about the scope of the invention" fail the definiteness requirement of 35 U.S.C. § 112, ¶ 2. *Nautilus, Inc. v. Biosig Instruments, Inc.*, 572 U.S. 898, 901 (2014). Samsung contends that, at least under Headwater's actual and/or apparent application of the claims, the Asserted Claims of the Asserted Patents fail to distinctly claim what the inventors regard as their alleged invention.

Each of the asserted claims are invalid as indefinite under 35 U.S.C. § 112 because they fail to particularly point out and distinctly claim the subject matter which the applicant regards as his invention. In particular, the following limitations, read in light of the intrinsic evidence, fail to inform those skilled in the art with reasonable certainty about the scope of the claimed inventions:

1. ’701 patent

- **Claim 2:** “a user of the device is not directly interacting with that application or perceiving any benefit from that application.”

2. ’184 patent

- **Claim 2:** “the user of the device is directly interacting with that application or perceiving any benefit from that application.”

3. ’578 patent

- **Claim 2:** “a user of the device is directly interacting with the particular application or perceiving any benefit from the particular application.”

4. ’445 patent

- **Claim 2:** “the user of the device is directly interacting with that application or perceiving any benefit from that application.”

5. ’976 patent

- **Claim 2:** “the user of the device is not directly interacting with that application or perceiving any benefit from that application.”

6. ’433 patent

- **Claim 3:** “the user of the device is directly interacting with that application or perceiving any benefit from that application.”

7. ’544 patent

- **Claim 2:** “the user of the device is directly interacting with that application or perceiving any benefit from that application.”

Samsung’s investigation of grounds of invalidity based upon indefiniteness is ongoing, and Samsung reserves the right to supplement these Contentions.

VII. INVALIDITY UNDER 35 U.S.C. § 101

Samsung contends that all claims of the Asserted Patents are invalid under 35 U.S.C. § 101 because the claims are not directed to patent-eligible subject matter. Samsung’s contentions that the Asserted Claims are invalid under 35 U.S.C. § 101 do not constitute, and should not be interpreted as, admissions regarding the construction or scope of the claims of the Asserted Patents, or that any of the claims of the Asserted Patents are not anticipated or rendered obvious by prior art.

35 U.S.C. § 101 provides that “[w]hoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor[.]” Because abstract ideas form the “basic tools of scientific and technological work,” they are unpatentable subject matter under 35 U.S.C. § 101. *Alice Corp. v. CLS Bank Int’l*, 573 U.S. 208, 216 (2014). The Supreme Court provided a two-part test for assessing patent eligibility under Section 101. *See generally id.*

Under the first step of *Alice* (“Step One”), a court must decide whether the claims are directed to ineligible subject matter, such as an abstract idea. *RecogniCorp, LLC v. Nintendo Co.*, 855 F.3d 1322, 1326 (Fed. Cir. 2017); *Internet Patents Corp. v. Active Network, Inc.*, 790 F.3d 1343, 1346 (Fed. Cir. 2015). To do so, a court examines the claims to determine whether their “character as a whole,” or their “focus,” is an abstract idea. *Elec. Power Grp., LLC v. Alstom S.A.*, 830 F.3d 1350, 1353 (Fed. Cir. 2016). This examination entails “identify[ing] and defin[ing] whatever fundamental concept appears wrapped up" in the claims. *Accenture Glob. Servs., GmbH v. Guidewire Software, Inc.*, 728 F.3d 1336, 1341 (Fed. Cir. 2013) (internal quotation marks and citations omitted). Once ascertained, the court then determines whether that character is "directed to excluded subject matter," such as an abstract idea. *Enfish, LLC v. Microsoft Corp.*, 822 F.3d 1327, 1335 (Fed. Cir. 2016) (internal citation omitted). This inquiry often asks whether the claims' character is directed to "a specific means or method" for improving technology or whether it is simply directed to an abstract end-result. *RecogniCorp*, 855 F.3d at 1326. If the claims are not directed to an abstract idea, the inquiry ends. *Thales Visionix Inc. v. United States*, 850 F.3d 1343, 1349 (Fed. Cir. 2017). To resolve this question, "it is often helpful to ask whether the claims are directed to 'an improvement in the functioning of a computer,' or merely 'adding conventional computer components to well-known business practices.'" *Affinity Labs. of Texas, LLC v. Amazon.com Inc.*, 838 F.3d 1266, 1270 (Fed. Cir. 2016) (quoting *Enfish, LLC v. Microsoft Corp.*, 822 F.3d 1327, 1338 (Fed. Cir. 2016)). "Generalized steps to be performed on a computer using conventional computer activity are abstract[.]" *RecogniCorp*, 855 F.3d at 1326 (internal quotation marks omitted). And "[c]laims directed to generalized steps to be performed on a computer using conventional computer activity are not patent eligible." *Affinity Labs.*, 838 F.3d at 1270 (citing *Internet Patents Corp. v. Active Network, Inc.*, 790 F.3d 1343, 1348-49 (Fed. Cir. 2015)).

If the claims, as here, are directed to one or more abstract ideas, then the court advances to the second step of *Alice* ("Step Two"), where the claim elements must be scrutinized "both individually and 'as an ordered combination' to determine whether the additional elements 'transform the nature of the claim' into a patent-eligible application." *Enfish*, 822 F.3d at 1354 (*quoting Alice*, 573 U.S. at 217). At Step Two, the court searches for an "'inventive concept' sufficient to 'transform the nature of the claim into a patent-eligible application.'" *RecogniCorp*, 855 F.3d at 1327 (quoting *Alice*, 573 U.S. at 217). To save a patent at Step Two, an inventive concept must be evident in the claims. *See Alice*, 573 U.S. at 221; *Synopsys, Inc. v. Mentor Graphics Corp.*, 839 F.3d 1138, 1149 (Fed. Cir. 2016). The "inventive concept" must also "involve more than performance of well-understood, routine, and conventional activities previously known to the industry." *Berkheimer v. HP Inc.*, 881 F.3d 1360, 1367 (Fed. Cir. 2018); *Mayo Collaborative Servs. v. Prometheus Labs., Inc.*, 566 U.S. 66, 73 (2012); *Certain Elec. Devices*, 2020 WL 6441422 at *4. The Federal Circuit's "precedent is clear that merely adding computer functionality to increase the speed or efficiency of the process does not confer patent eligibility on an otherwise abstract idea." *Ericsson Inc. v. TCL Commc'n Tech. Holdings Ltd.*, 955 F.3d 1317, 1330 (Fed. Cir. 2020) (citing *Intellectual Ventures I LLC v. Capital One Bank (USA)*, 792 F.3d 1363, 1370 (Fed. Cir. 2015)). In addition, limitations that simply restrict the invention to a single field of use or merely add "token post-solution" requirements also fail to impart patentability. *Bilski v. Kappos*, 561 U.S. 593, 612 (2010).

All Asserted Claims are invalid under 35 U.S.C. § 101 because they fail to claim patent-eligible subject matter, including the following claims:

- Claims 1-25 of the '701 patent;
- Claims 1-20 of the '184 patent;
- Claims 1-22 of the '578 patent;
- Claims 1-20, 22-26 of the '445 patent;
- Claims 1-17 of the '224 patent;
- Claims 1-29 of the '976 patent;
- Claims 1-20 of the '433 patent;
- Claims 1-8 and 11-23 of the '544 patent; and

- Claims 1-14 and 16-20 of the ’773 patent.

A. Alice Step One: The Asserted Claims Embody Abstract Concepts

At Step One, the court must “identify and define whatever fundamental concept appears wrapped up in the claim.” *Accenture Glob. Servs.*, 728 F.3d at 1341.

All asserted independent claims are substantially similar and related to optimizing and/or prioritizing traffic and resources of mobile phones. At the most basic level, the Asserted Claims are linked to the same abstract idea of prioritizing information flow and resource allocation, which is a long prevalent human activity. Because the Asserted Claims fail to offer anything more than using generic pre-existing computer functionality to carry out the abstract idea of prioritizing information flow and resource allocation, the Asserted Claims are abstract at Step One. *See, e.g.*, ’976 pat. at 2:66-3:8 (“[T]hese implementations, or any other form that the invention may take, may be referred to as ***techniques***. . . . Unless stated otherwise, a component such as a processor or a memory described as being configured to perform a task may be implemented as a ***general component*** that is temporarily configured to perform the task…”).

B. Alice Step Two: The Asserted Claims Do Not Recite an Inventive Concept

For a claim to be patent eligible under *Alice* Step Two, “an inventive concept must be evident in the claims,” *RecogniCorp*, 855 F.3d at 1327, and it must provide “significantly more” than the abstract idea itself, *BSG Tech LLC v. Buyseasons, Inc.*, 899 F.3d 1281, 1289-90 (Fed. Cir. 2018). In performing this analysis, a court must scrutinize the claim elements “both individually and ‘as an ordered combination’ to determine whether the additional elements ‘transform the nature of the claim’ into a patent-eligible application.” *Enfish*, 822 F.3d at 1354 (quoting *Alice*, 573 U.S. at 217). Elements that are “well-understood, routine, conventional,” or “purely functional” cannot confer patent-eligibility. *Alice*, 573 U.S. at 225-26 (citation omitted). Admittedly, the claims relate to the same “general communication device” described in generic terms in the specification and the claimed process carried out on a generic processor is performed using well-known standards. Thus, there is no particularized solution to any problem, and nothing that can pass Step 2 of *Alice*.

The Asserted Claims recite only conventional, well known components and functionalities, and do not recite any purportedly novel arrangement of those components. As set forth in Exhibits A-01 through I-11, Exhibits A-A through I-I, and/or through AAPA, individually and/or collectively, these features were well known, routine, and/or conventional, and known in the prior art as set forth herein.

VIII. Document Production

Pursuant to Patent Rule 3-4, Samsung is concurrently producing the prior art identified in these Invalidity Contentions, but Samsung is not required to produce the prior art in the file histories of the Asserted Patents.

In addition, based on investigations to date, Samsung is concurrently producing and/or making available for inspection source code, specifications, schematics, flow charts, artwork, formulas, or other documentation sufficient to show the operation of any aspects or elements of the Accused Instrumentalities identified by Headwater in its P.R. 3-1(c) chart.

Samsung reserves the right to supplement these productions with additional documentation, in accordance with the Federal Rules of Civil Procedure, the Local Rules, the Court's orders and other applicable rules and statutes.

Dated: May 9, 2023

Respectfully submitted,

By: */s/ Jared Hartzman*
Ruffin B. Cordell
TX Bar No. 04820550
Michael J. McKeon
DC Bar No. 459780

mckeon@fr.com
Jared Hartzman (*pro hac vice*)
DC Bar No. 1034255
hartzman@fr.com
Joshua Carrigan (*pro hac vice*)
VA Bar No. 96911
carrigan@fr.com
FISH & RICHARDSON P.C.
1000 Maine Avenue, SW, Ste 1000
Washington, D.C. 20024
Telephone: (202) 783-5070
Facsimile: (202) 783-2331

Thad C. Kodish
GA Bar No. 427603
tkodish@fr.com
Benjamin K. Thompson
GA Bar No. 633211
bthompson@fr.com
Nicholas A. Gallo (*pro hac vice*)
GA Bar No. 546590
gallo@fr.com
Steffen Lake (*pro hac vice*)
GA Bar No. 512272
lake@fr.com
FISH & RICHARDSON P.C.
1180 Peachtree St. NE, Fl. 21
Atlanta, GA 30309
Telephone: (404) 892-5005
Facsimile: (404) 892-5002

Leonard E. Davis
TX Bar No. 05521600
ldavis@fr.com
Andria Rae Crisler
TX Bar No. 24093792
crisler@fr.com
FISH & RICHARDSON P.C.
1717 Main Street, Suite 5000
Dallas, TX 75201
Telephone: (214)747-5070
Facsimile: (214) 747-2091

John-Paul R. Fryckman (*pro hac vice*)
CA Bar No. 317591
FISH & RICHARDSON P.C.

12860 El Camino Real, Ste. 400
San Diego, CA 92130
Telephone: (858) 678-5070
Facsimile: (858) 678-5099

Melissa R. Smith
State Bar No. 24001351
Melissa@gillamsmithlaw.com
GILLAM & SMITH, LLP
303 South Washington Avenue
Marshall, Texas 75670
Telephone: (903) 934-8450
Facsimile: (903) 934-9257

Andrew Thompson ("Tom") Gorham
State Bar No. 24012715
tom@gillamsmithlaw.com
GILLAM & SMITH, LLP
102 N. College, Ste. 800
Tyler, Texas 75702
Telephone: (903) 934-8450
Facsimile: (903) 934-9257

Grant K. Schmidt
Texas Bar No. 24084579
gschmidt@hilgersgraben.com
Jon Hyland
Texas Bar No. 24046131
jhyland@hilgersgraben.com
Theo Kwong
Texas Bar No. 24087871
tkwong@hilgersgraben.com
HILGERS GRABEN PLLC
7859 Walnut Hill Lane, Suite 335
Dallas, Texas 75230
Telephone: (469) 751-2819

Attorneys for Defendants
Samsung Electronics Co., Ltd. and
Samsung Electronics America, Inc.

CERTIFICATE OF SERVICE

The undersigned hereby certifies that a true and correct copy of the foregoing was served on counsel of record for Plaintiff via electronic mail on May 9, 2023.

/s/ Jared Hartzman
Jared Hartzman

Exhibit A-10

Based on Headwater's apparent positions as to the scope of the patent's claims, as best they can be deciphered, the reference(s) charted below anticipate(s) or at least render(s) obvious the identified claims. The portions of the prior art ~~reference~~ system documents cited below are not exhaustive but are exemplary in nature. Additional citations may be found in the cover pleading.

This disclosure is not an admission that Samsung concedes any claim construction implied or suggested by Headwater's apparent positions as to the scope of the patent's claims, nor is it an admission by Samsung that any of its products are covered by or infringe the patent's claims, particularly when they are properly construed and applied. Samsung is not taking any claim construction positions through this disclosure, including whether the preamble is a limitation.

Samsung reserves the right to rely on additional citations or sources of evidence that also may be applicable, or that may become applicable in light of claim construction, changes in Headwater's infringement contentions, and/or information obtained during discovery as the case progresses. Samsung further reserves the right to amend or supplement this claim chart at a later date as more fully set forth in the Invalidity Contentions. For example, Defendants are currently in the process of taking discovery from non-parties including Nokia, HMD, Citrix, Google, Apple, and Microsoft. Accordingly, Defendants reserve the right to modify, amend, and/or supplement these contentions as information becomes available from non-parties.

Android is mobile device operating system that was initially released in September 2008. Applications (or "apps") can be installed on mobile devices that run Android. Any mobile device that predates the '701 patent, running an Android version with one or more apps that also predate the '701 patent, qualifies as prior art under at least pre-AIA 35 U.S.C. §§ 102(a)/(b). Such a device was known, used, offered for sale, and/or sold in the United States before the '701 patent.

Exemplary mobile devices that predate the '701 patent and were publicly available before the earliest possible priority date include:[1]

- HTC Dream/T-Mobile G1 (released September 2008)

[1] *See, e.g.*, SAMSUNG_PRIORART0000001-334; SAMSUNG_PRIORART0005174-76; SAMSUNG_PRIORART0005177-317; SAMSUNG_PRIORART0005416-19; SAMSUNG_PRIORART0005420-23; SAMSUNG_PRIORART0005424-28; SAMSUNG_PRIORART0005429-44; SAMSUNG_PRIORART0005445-48; SAMSUNG_PRIORART0005449-52; SAMSUNG_PRIORART0005453-57; SAMSUNG_PRIORART0005458-71; SAMSUNG_PRIORART0005472-77; SAMSUNG_PRIORART0005478-84; SAMSUNG_PRIORART0005485-86; SAMSUNG_PRIORART0005488-5624.

- Samsung GT-I7500 Galaxy (released June 2009)
- Nexus One (released January 2010)

Exemplary Android versions that predate the ’701 patent and were publicly available before the earliest possible priority date include:[2]

- Android 1.0 (released September 2008)
- Android 1.1 (released February 2009)
- Android Cupcake (1.5) (released April 2009)
- Android Donut (1.6) (released September 2009)
- Android Eclair (2.0, 2.0.1, 2.1) (released October 2009 - January 2010)
- Android Froyo (2.2) (released May 20, 2010)

[2] See, e.g., SAMSUNG_PRIORART0003998; SAMSUNG_PRIORART0004085; SAMSUNG_PRIORART0004081; SAMSUNG_PRIORART0004086; SAMSUNG_PRIORART0004083; SAMSUNG_PRIORART0004084; GOOG-HEADWATER-000000001-123; HDWTR-GOOG00001-GOOG00013; SAMSUNG_PRIORART0005042; SAMSUNG_PRIORART0005062; SAMSUNG_PRIORART0005350; SAMSUNG_PRIORART0005351; SAMSUNG_PRIORART0005352; SAMSUNG_PRIORART0005353; SAMSUNG_PRIORART0005354; SAMSUNG_PRIORART0005355; SAMSUNG_PRIORART0005356; SAMSUNG_PRIORART0005357; SAMSUNG_PRIORART0005358; SAMSUNG_PRIORART0005359; SAMSUNG_PRIORART0005360; SAMSUNG_PRIORART0005361; SAMSUNG_PRIORART0005362; SAMSUNG_PRIORART0005363; SAMSUNG_PRIORART0005364; SAMSUNG_PRIORART0005046; SAMSUNG_PRIORART0005043; SAMSUNG_PRIORART0005044; SAMSUNG_PRIORART0005045; SAMSUNG_PRIORART0005054; SAMSUNG_PRIORART0005055; SAMSUNG_PRIORART0005056; SAMSUNG_PRIORART0005057; SAMSUNG_PRIORART0005058; SAMSUNG_PRIORART0005059; SAMSUNG_PRIORART0005060; SAMSUNG_PRIORART0005061; SAMSUNG_PRIORART0005318; SAMSUNG_PRIORART0005398; SAMSUNG_PRIORART0005047-53; SAMSUNG_PRIORART0005063-65; SAMSUNG_PRIORART0005066-74; SAMSUNG_PRIORART0005075-135; SAMSUNG_PRIORART0005136-52; SAMSUNG_PRIORART0005153-70; SAMSUNG_PRIORART0005171-73; SAMSUNG_PRIORART0005319-49; SAMSUNG_PRIORART0005365-74; SAMSUNG_PRIORART0005375-83; SAMSUNG_PRIORART0005384-94; SAMSUNG_PRIORART0005395-97; SAMSUNG_PRIORART0005487.~~.~~

Android included files[3] such as ConnectivityManager, NetworkInfo, NetworkStateTracker, ThrottleManager, TrafficStats, ConnectivityManagerMobileTest, Socket, SocketTest, Power, PowerManager, PowerManagerTest, BatteryManager, and BatteryStats.

-

Exemplary apps that predate the '701 patent and were publicly available before the earliest possible priority date include:[4]

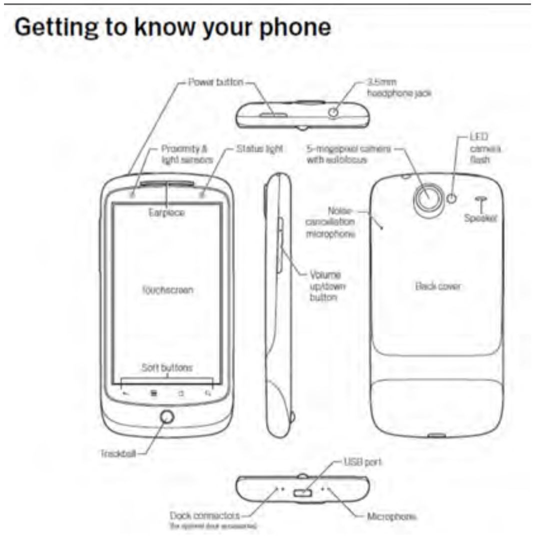
- JuiceDefender (released January 2010) and its associated add-on application, UltimateJuice (collectively "JuiceDefender App")
- GreenPower (released March 2010)

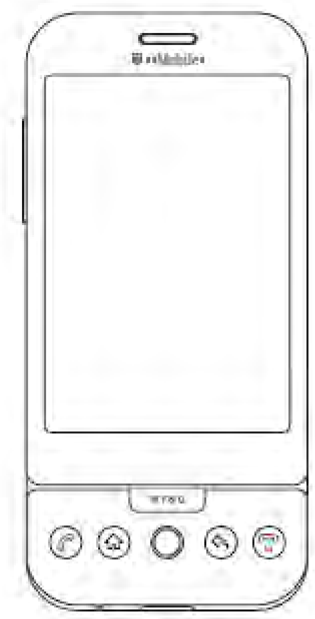
As specific examples, an HTC Dream/T-Mobile G1, Samsung GT-I7500 Galaxy, or a Nexus One mobile device running any of Android versions 1.0- 2.2 by itself, or with the JuiceDefender or GreenPower applications installed, ~~As a specific example, a Nexus One mobile device running Android Froyo 2.2 with the JuiceDefender application installed~~ qualifies as prior art under at least pre-AIA 35 U.S.C. §§ 102(a)/(b). This device was known, used, offered for sale, and/or sold in the United States on or before May 20, 2010. At least the various documents cited in this claim chart describe the functionality of this device.

To the extent it is argued that Android Device with One or More Apps does not disclose or include each and every asserted claim limitation, either expressly or inherently, it would have been obvious to a POSITA to incorporate any of the teachings from the references identified in Exhibits A-01 through A-11, and A-A (whose exemplary citations for each limitation are incorporated herein) into Android Device with One or More Apps. Indeed, it would have been obvious to make such combinations and a POSITA would have had reason and motivation to make such combinations at least for reasons described herein and in the cover pleading.

'701 Claims	Android Device with One or More Apps
[1 pre] A wireless end-user device, comprising:	To the extent the preamble is a limitation, Android Device with One or More Apps discloses and/or renders obvious this element. For example, see the following passages and/or figures, as well as related disclosures:

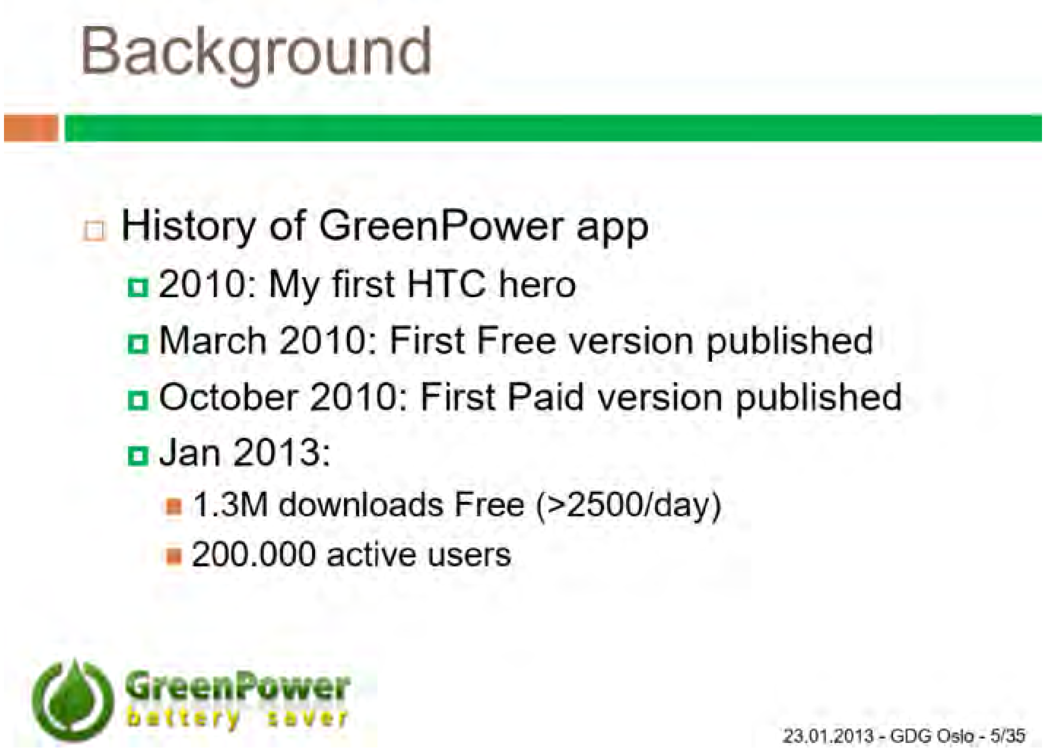
[3] The files listed are Java source files, so the filenames are ConnectivityManager.java, NetworkInfo.java, etc., except as noted.
[4] See, e.g., SAMSUNG_PRIORART0000335-SAMSUNG_PRIORART0000383; POUZERATE0000001-POUZERATE0000261.

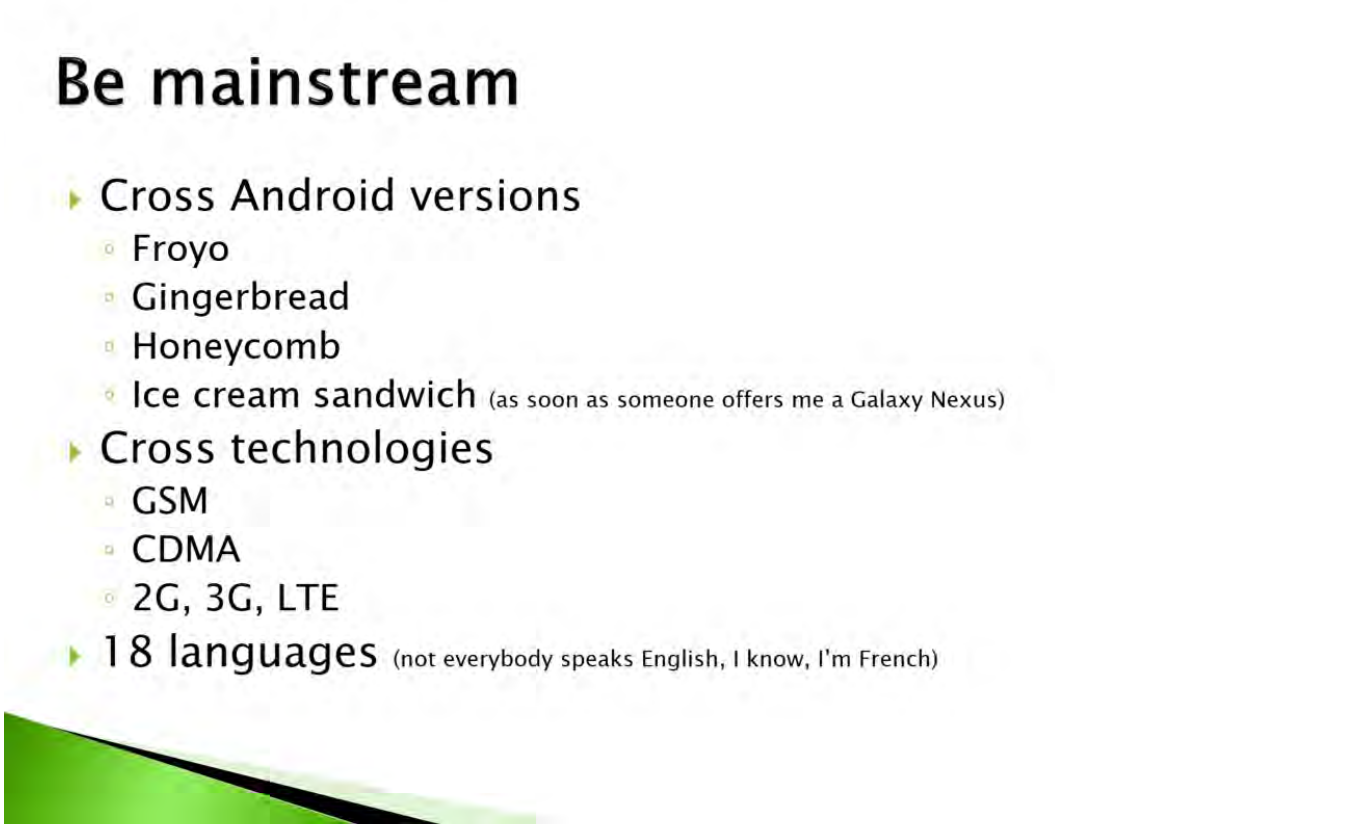
'701 Claims	Android Device with One or More Apps
	Nexus One The Nexus One is an example of an Android smartphone. *See*, *e.g.*, SAMSUNG_PRIORART0000001 (Nexus) at 17:  Getting to know your phone Power button 3.5mm headphone jack Proximity & light sensors Status light 5-megapixel camera with autofocus LED camera flash Earpiece Noise-cancellation microphone Speaker Touchscreen Volume up/down button Back cover Soft buttons Trackball USB port Dock connectors (for optional dock accessories) Microphone **HTC Dream / T-Mobile G1** The HTC Dream / T-Mobile G1 is an example of an Android smartphone. SAMSUNG_PRIORART0005184

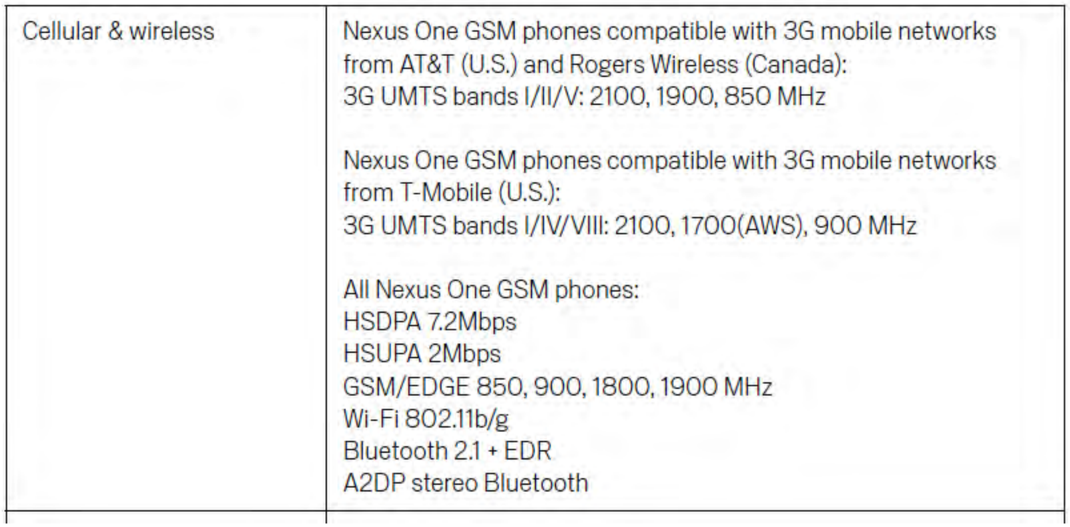
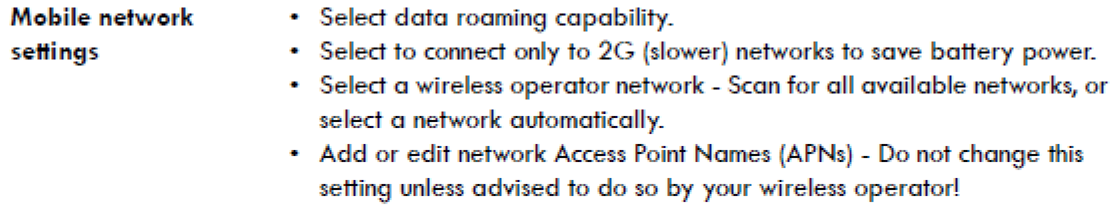
’701 Claims	Android Device with One or More Apps
	 T-Mobile G1 SAMSUNG_PRIORART0005177 Android™ mobile technology platform R1.0 Document Rev 08 - September 8, 2008 Copyright 2008 © Google, Inc. All rights reserved. **Samsung GT-I7500 Galaxy**

'701 Claims	Android Device with One or More Apps
	The Samsung GT-I7500 Galaxy is an example of an Android smartphone. SAMSUNG_PRIORART0005494 at pgs. PDF 2, 19: SAMSUNG OK Earpiece Volume key Confirm key Menu key Dial key Proximity sensor Touch screen 4-way navigation key Back key Home key Power key Mouthpiece **JuiceDefender App** JuiceDefender is a mobile application (or "app") intended to run on a mobile device, such an Android smartphone. The Nexus One is an example of an Android smartphone capable of running JuiceDefender.

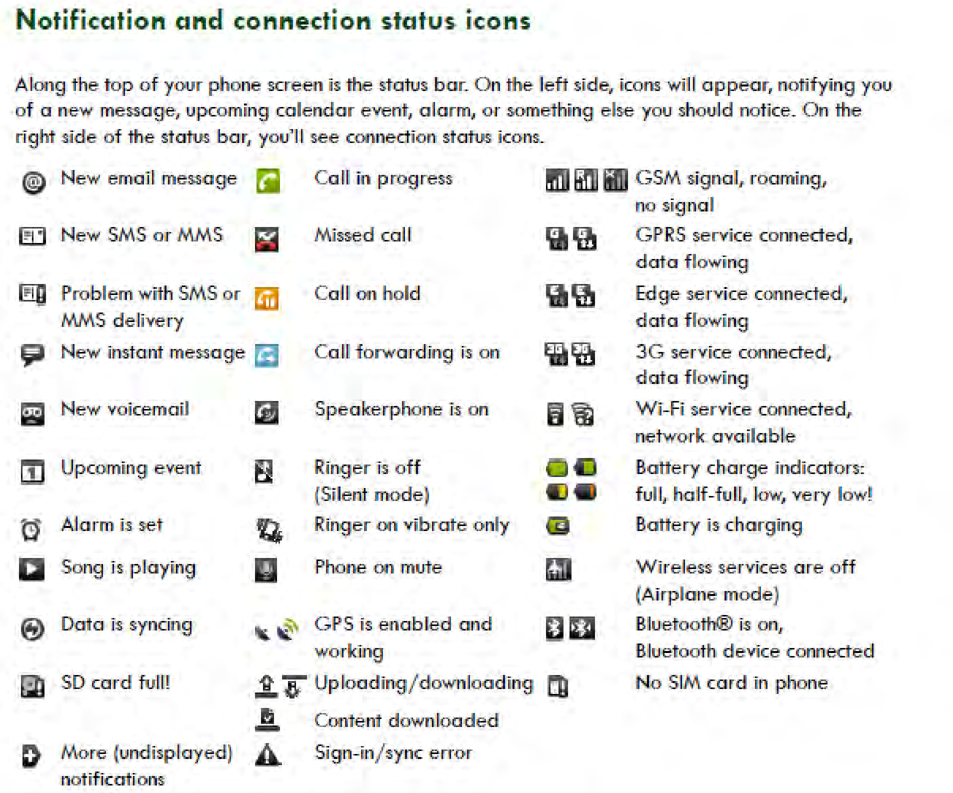
'701 Claims	**Android Device with One or More Apps**
	SAMSUNG_PRIORART0000379 (Latedroid) ("JuiceDefender saves battery power (lots of it!) by controlling the device data connection and/or WiFi … You can schedule regular APN/WiFi activation to let background data sync occur and have APN/WiFi enabled while the screen is on. It also helps in minimizing distractions.") SAMSUNG_PRIORART0000361 (Purdy) ("Android: Most phones don't make it easy to switch cellular data connection on and off, even if doing so really helps save your battery. JuiceDefender toggles wireless data and Wi-Fi on and off every so often to preserve power.") **GreenPower App** GreenPower is a mobile application (or "app") intended to run on a mobile device, such an Android smartphone. The Nexus One is an example of an Android smartphone capable of running GreenPower. *See, e.g.*, POUZERATE0000015 (GDG Oslo) at 5:

'701 Claims	Android Device with One or More Apps
	 Background □ History of GreenPower app ▫ 2010: My first HTC hero ▫ March 2010: First Free version published ▫ October 2010: First Paid version published ▫ Jan 2013: ■ 1.3M downloads Free (>2500/day) ■ 200.000 active users GreenPower battery saver 23.01.2013 - GDG Oslo - 5/35 *See, e.g.*, POUZERATE0000002 (App Circus) at 9:

'701 Claims	Android Device with One or More Apps
	 Be mainstream - Cross Android versions - Froyo - Gingerbread - Honeycomb - Ice cream sandwich (as soon as someone offers me a Galaxy Nexus) - Cross technologies - GSM - CDMA - 2G, 3G, LTE - 18 languages (not everybody speaks English, I know, I'm French) *See also* GOOG-HEADWATER-000000001-123.
[1a] a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: **<u>Nexus One</u>**

'701 Claims	Android Device with One or More Apps
device and at least one WWAN, when configured for and connected to the WWAN;	*See, e.g.*, SAMSUNG_PRIORART0000001 (Nexus) at 332:  Cellular & wireless Nexus One GSM phones compatible with 3G mobile networks from AT&T (U.S.) and Rogers Wireless (Canada): 3G UMTS bands I/II/V: 2100, 1900, 850 MHz Nexus One GSM phones compatible with 3G mobile networks from T-Mobile (U.S.): 3G UMTS bands I/IV/VIII: 2100, 1700(AWS), 900 MHz All Nexus One GSM phones: HSDPA 7.2Mbps HSUPA 2Mbps GSM/EDGE 850, 900, 1800, 1900 MHz Wi-Fi 802.11b/g Bluetooth 2.1 + EDR A2DP stereo Bluetooth **HTC Dream / T-Mobile G1** SAMSUNG_PRIORART0005202  **Mobile network settings** • Select data roaming capability. • Select to connect only to 2G (slower) networks to save battery power. • Select a wireless operator network - Scan for all available networks, or select a network automatically. • Add or edit network Access Point Names (APNs) - Do not change this setting unless advised to do so by your wireless operator! SAMSUNG_PRIORART0005200

'701 Claims	Android Device with One or More Apps



Notification and connection status icons

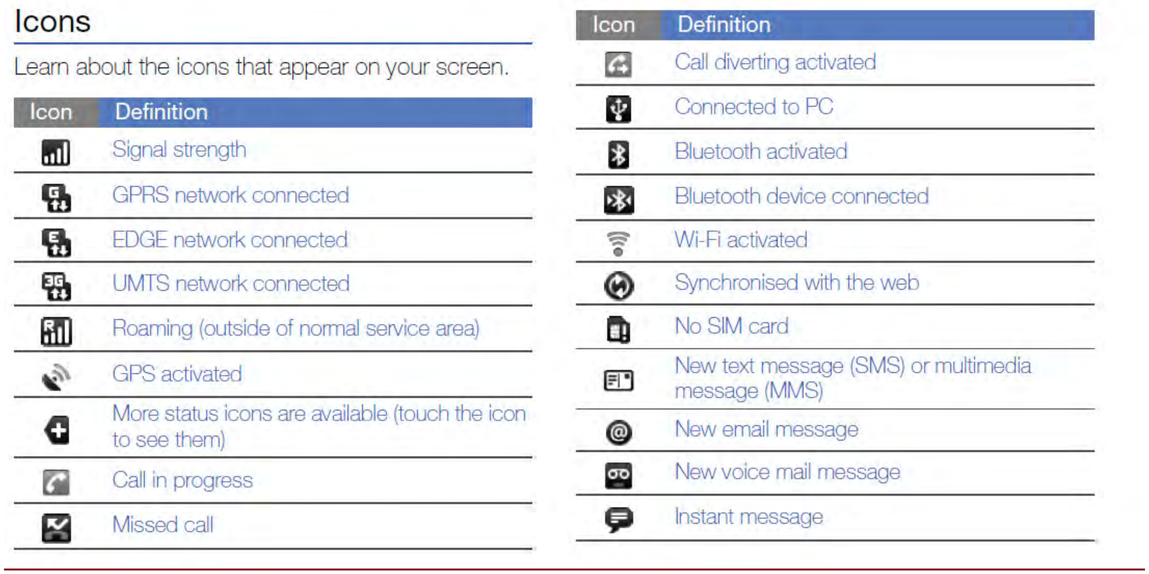
Along the top of your phone screen is the status bar. On the left side, icons will appear, notifying you of a new message, upcoming calendar event, alarm, or something else you should notice. On the right side of the status bar, you'll see connection status icons.

New email message	Call in progress	GSM signal, roaming, no signal
New SMS or MMS	Missed call	GPRS service connected, data flowing
Problem with SMS or MMS delivery	Call on hold	Edge service connected, data flowing
New instant message	Call forwarding is on	3G service connected, data flowing
New voicemail	Speakerphone is on	Wi-Fi service connected, network available
Upcoming event	Ringer is off (Silent mode)	Battery charge indicators: full, half-full, low, very low!
Alarm is set	Ringer on vibrate only	Battery is charging
Song is playing	Phone on mute	Wireless services are off (Airplane mode)
Data is syncing	GPS is enabled and working	Bluetooth® is on, Bluetooth device connected
SD card full!	Uploading/downloading	No SIM card in phone
	Content downloaded	
More (undisplayed) notifications	Sign-in/sync error	

Samsung GT-I7500 Galaxy

SAMSUNG_PRIORART0005494 at PDF pg. 21:

’701 Claims	Android Device with One or More Apps



Icons

Learn about the icons that appear on your screen.

Icon	Definition
📶	Signal strength
G	GPRS network connected
E	EDGE network connected
3G	UMTS network connected
R	Roaming (outside of normal service area)
	GPS activated
+	More status icons are available (touch the icon to see them)
✆	Call in progress
	Missed call

Icon	Definition
	Call diverting activated
	Connected to PC
	Bluetooth activated
	Bluetooth device connected
	Wi-Fi activated
	Synchronised with the web
	No SIM card
	New text message (SMS) or multimedia message (MMS)
@	New email message
oo	New voice mail message
	Instant message

<u>JuiceDefender App</u>

SAMSUNG_PRIORART0000379 (Latedroid) (“JuiceDefender saves battery power (lots of it!) by controlling the device data connection and/or WiFi … You can schedule regular APN/WiFi activation to let background data sync occur and have APN/WiFi enabled while the screen is on. It also helps in minimizing distractions.”).

SAMSUNG_PRIORART0000351 (Configuration-Translated) (“**APN:** activates / deactivates the APN connection, in its submenu we find the **MMS** button that activated configures the reception of MMS in the same way that we have the APN and **Prefer Wifi** that activated will try to connect first to this and if to five seconds does not find an available network will activate the APN. **WIFI:** activates / deactivates

'701 Claims	Android Device with One or More Apps
	the WIFI connection, in its submenu we find the following buttons, **Auto Disable** turns off the wifi in the case of not finding a network available to save battery, in case it is deactivated we will have to activate it manually; **Enable on Schedule / Peak / Screen will** activate the wifi as we have configured those buttons that we will see below.") SAMSUNG_PRIORART0000361 (Purdy) ("Android: Most phones don't make it easy to switch cellular data connection on and off, even if doing so really helps save your battery. JuiceDefender toggles wireless data and Wi-Fi on and off every so often to preserve power."). SAMSUNG_PRIORART0000335 (Ruddock) ("Juice Defender is a battery conservation app. It uses various triggers, rules, and timers to control how often your device utilizes 3G/EDGE APN's (data connections) as well as WiFi. These data connections are the number one drainers of battery life when your phone is idle, so Juice Defender allows you to decide when, where, and how often you want them to be active."). **Android 1.0** **GOOG-HEADWATER-00000040, SAMSUNG_PRIORART0005487, ConnectivityManager** /** * Class that answers queries about the state of network connectivity. It also * notifies applications when network connectivity changes. Get an instance * of this class by calling * {@link android.content.Context#getSystemService(String) Context.getSystemService(Context.CONNECTIVITY_SERVICE)}. * <p> * The primary responsibilities of this class are to: * <ol> * <li>Monitor network connections (Wi-Fi, GPRS, UMTS, etc.)</li> * <li>Send broadcast intents when network connectivity changes</li> * <li>Attempt to "fail over" to another network when connectivity to a network

’701 Claims	Android Device with One or More Apps

	* is lost</li>
	* <li>Provide an API that allows applications to query the coarse-grained or fine-grained
	* state of the available networks</li>
	* </ol>
	*/
	
	    public static final int TYPE_MOBILE = 0;
	    public static final int TYPE_WIFI = 1;
	    public static final int DEFAULT_NETWORK_PREFERENCE = TYPE_WIFI;
	
	    static public boolean isNetworkTypeValid(int networkType) {
	        return networkType == TYPE_WIFI || networkType == TYPE_MOBILE;
	    }
	
	    /** {@hide} */
	    public boolean setRadios(boolean turnOn) {
	        try {
	            return mService.setRadios(turnOn);
	        } catch (RemoteException e) {
	            return false;
	        }
	    }
	    /** {@hide} */
	    public boolean setRadio(int networkType, boolean turnOn) {
	        try {
	            return mService.setRadio(networkType, turnOn);
	        } catch (RemoteException e) {
	            return false;
	        }
	    }

'701 Claims	Android Device with One or More Apps

	SAMSUNG_PRIORART0005487, NetworkInfo
	/**
	* Indicates whether network connectivity exists or is in the process
	* of being established. This is good for applications that need to
	* do anything related to the network other than read or write data.
	* For the latter, call {@link #isConnected()} instead, which guarantees
	* that the network is fully usable.
	* @return {@code true} if network connectivity exists or is in the process
	* of being established, {@code false} otherwise.
	*/
	public boolean isConnectedOrConnecting() {
	return mState == State.CONNECTED || mState == State.CONNECTING;
	}
	
	/**
	* Indicates whether network connectivity exists and it is possible to establish
	* connections and pass data.
	* @return {@code true} if network connectivity exists, {@code false} otherwise.
	*/
	public boolean isConnected() {
	return mState == State.CONNECTED;
	}
	/**
	* Indicates whether network connectivity is possible. A network is unavailable
	* when a persistent or semi-persistent condition prevents the possibility
	* of connecting to that network. Examples include
	* <ul>
	* <li>The device is out of the coverage area for any network of this type.</li>

'701 Claims	Android Device with One or More Apps

	     * <li>The device is on a network other than the home network (i.e., roaming), and
	     * data roaming has been disabled.</li>
	     * <li>The device's radio is turned off, e.g., because airplane mode is enabled.</li>
	     * </ul>
	     * @return {@code true} if the network is available, {@code false} otherwise
	     */
	    public boolean isAvailable() {
	        return mIsAvailable;
	    }
	
	    /**
	     * Sets if the network is available, ie, if the connectivity is possible.
	     * @param isAvailable the new availability value.
	     *
	     * {@hide}
	     */
	    public void setIsAvailable(boolean isAvailable) {
	        mIsAvailable = isAvailable;
	    }
	
	    public String getTypeName() {
	        switch (mNetworkType) {
	            case ConnectivityManager.TYPE_WIFI:
	                return "WIFI";
	            case ConnectivityManager.TYPE_MOBILE:
	                return "MOBILE";
	            default:
	                return "<invalid>";
	        }
	    }

'701 Claims	Android Device with One or More Apps

	SAMSUNG_PRIORART0005487, NetworkStateTracker
	/**
	* Turn the wireless radio off for a network.
	* @param turnOn {@code true} to turn the radio on, {@code false}
	*/
	public abstract boolean setRadio(boolean turnOn);
	
	/**
	* Returns an indication of whether this network is available for
	* connections. A value of {@code false} means that some quasi-permanent
	* condition prevents connectivity to this network.
	*/
	public abstract boolean isAvailable();
	
	SAMSUNG_PRIORART0005487, ConnectivityService
	/*
	* Create the network state trackers for Wi-Fi and mobile
	* data. Maybe this could be done with a factory class,
	* but it's not clear that it's worth it, given that
	* the number of different network types is not going
	* to change very often.
	*/
	if (DBG) Log.v(TAG, "Starting Wifi Service.");
	mWifiStateTracker = new WifiStateTracker(context, handler);
	WifiService wifiService = new WifiService(context, mWifiStateTracker);
	ServiceManager.addService(Context.WIFI_SERVICE, wifiService);
	// The WifiStateTracker should appear first in the list
	mNetTrackers[ConnectivityManager.TYPE_WIFI] = mWifiStateTracker;

’701 Claims	Android Device with One or More Apps

```
        mMobileDataStateTracker = new MobileDataStateTracker(context, handler);
        mNetTrackers[ConnectivityManager.TYPE_MOBILE] = mMobileDataStateTracker;

        mActiveNetwork = null;
        mNumDnsEntries = 0;
        mTestMode = SystemProperties.get("cm.test.mode").equals("true")
                && SystemProperties.get("ro.build.type").equals("eng");
        for (NetworkStateTracker t : mNetTrackers)
            t.startMonitoring();
        // Constructing this starts it too
        mWifiWatchdogService = new WifiWatchdogService(context, mWifiStateTracker);
    }

    /**
     * Make the state of network connectivity conform to the preference settings.
     * In this method, we only tear down a non-preferred network. Establishing
     * a connection to the preferred network is taken care of when we handle
     * the disconnect event from the non-preferred network
     * (see {@link #handleDisconnect(NetworkInfo)}).
     */

    /**
     * Ensure that a network route exists to deliver traffic to the specified
     * host via the specified network interface.
     * @param networkType the type of the network over which traffic to the specified
     * host is to be routed
     * @param hostAddress the IP address of the host to which the route is desired
     * @return {@code true} on success, {@code false} on failure
     */
```

'701 Claims	Android Device with One or More Apps
	Android 1.6 **SAMSUNG_PRIORART0005350, ConnectivityManager** /** * Class that answers queries about the state of network connectivity. It also * notifies applications when network connectivity changes. Get an instance * of this class by calling * {@link android.content.Context#getSystemService(String) Context.getSystemService(Context.CONNECTIVITY_SERVICE)}. * <p> * The primary responsibilities of this class are to: * <ol> * <li>Monitor network connections (Wi-Fi, GPRS, UMTS, etc.)</li> * <li>Send broadcast intents when network connectivity changes</li> * <li>Attempt to "fail over" to another network when connectivity to a network * is lost</li> * <li>Provide an API that allows applications to query the coarse-grained or fine-grained * state of the available networks</li> * </ol> */ @SdkConstant(SdkConstantType.BROADCAST_INTENT_ACTION) public static final String ACTION_BACKGROUND_DATA_SETTING_CHANGED = "android.net.conn.BACKGROUND_DATA_SETTING_CHANGED"; public static final int TYPE_MOBILE = 0; public static final int TYPE_WIFI = 1;

'701 Claims	Android Device with One or More Apps

	public static final int DEFAULT_NETWORK_PREFERENCE = TYPE_WIFI;
	
	static public boolean isNetworkTypeValid(int networkType) {
	return networkType == TYPE_WIFI || networkType == TYPE_MOBILE;
	}
	
	public void setNetworkPreference(int preference) {
	try {
	mService.setNetworkPreference(preference);
	} catch (RemoteException e) {
	}
	}
	
	/** {@hide} */
	public boolean setRadio(int networkType, boolean turnOn) {
	try {
	return mService.setRadio(networkType, turnOn);
	} catch (RemoteException e) {
	return false;
	}
	}
	
	/**
	* Returns the value of the setting for background data usage. If false,
	* applications should not use the network if the application is not in the
	* foreground. Developers should respect this setting, and check the value
	* of this before performing any background data operations.
	* <p>
	* All applications that have background services that use the network
	* should listen to {@link #ACTION_BACKGROUND_DATA_SETTING_CHANGED}.

'701 Claims	Android Device with One or More Apps

```
     *
     * @return Whether background data usage is allowed.
     */
    public boolean getBackgroundDataSetting() {
        try {
            return mService.getBackgroundDataSetting();
        } catch (RemoteException e) {
            // Err on the side of safety
            return false;
        }
    }

    /**
     * Sets the value of the setting for background data usage.
     *
     * @param allowBackgroundData Whether an application should use data while
     *            it is in the background.
     *
     * @attr ref android.Manifest.permission#CHANGE_BACKGROUND_DATA_SETTING
     * @see #getBackgroundDataSetting()
     * @hide
     */
    public void setBackgroundDataSetting(boolean allowBackgroundData) {
        try {
            mService.setBackgroundDataSetting(allowBackgroundData);
        } catch (RemoteException e) {
        }
    }
```

SAMSUNG_PRIORART0005350, NetworkInfo

'701 Claims	Android Device with One or More Apps

	/**
	 * Indicates whether network connectivity is possible:
	 */
	 private boolean mIsAvailable;
	
	/**
	 * Return a human-readable name describe the type of the network,
	 * for example "WIFI" or "MOBILE".
	 * @return the name of the network type
	 */
	 public String getTypeName() {
	 return mTypeName;
	 }
	
	 /**
	 * Indicates whether network connectivity exists or is in the process
	 * of being established. This is good for applications that need to
	 * do anything related to the network other than read or write data.
	 * For the latter, call {@link #isConnected()} instead, which guarantees
	 * that the network is fully usable.
	 * @return {@code true} if network connectivity exists or is in the process
	 * of being established, {@code false} otherwise.
	 */
	 public boolean isConnectedOrConnecting() {
	 return mState == State.CONNECTED || mState == State.CONNECTING;
	 }
	
	/**
	 * Indicates whether network connectivity is possible. A network is unavailable

'701 Claims	Android Device with One or More Apps

	* when a persistent or semi-persistent condition prevents the possibility
	* of connecting to that network. Examples include
	* <ul>
	* <li>The device is out of the coverage area for any network of this type.</li>
	* <li>The device is on a network other than the home network (i.e., roaming), and
	* data roaming has been disabled.</li>
	* <li>The device's radio is turned off, e.g., because airplane mode is enabled.</li>
	* </ul>
	* @return {@code true} if the network is available, {@code false} otherwise
	*/
	public boolean isAvailable() {
	return mIsAvailable;
	}
	
	/**
	* Indicates whether the device is currently roaming on this network.
	* When {@code true}, it suggests that use of data on this network
	* may incur extra costs.
	* @return {@code true} if roaming is in effect, {@code false} otherwise.
	*/
	public boolean isRoaming() {
	return mIsRoaming;
	}
	
	SAMSUNG_PRIORART0005350, ConnectivityService
	
	/*
	* Create the network state trackers for Wi-Fi and mobile
	* data. Maybe this could be done with a factory class,
	* but it's not clear that it's worth it, given that

'701 Claims	Android Device with One or More Apps

	* the number of different network types is not going
	     * to change very often.
	     */
	    if (DBG) Log.v(TAG, "Starting Wifi Service.");
	    mWifiStateTracker = new WifiStateTracker(context, handler);
	    WifiService wifiService = new WifiService(context, mWifiStateTracker);
	    ServiceManager.addService(Context.WIFI_SERVICE, wifiService);
	    mNetTrackers[ConnectivityManager.TYPE_WIFI] = mWifiStateTracker;
	
	    mMobileDataStateTracker = new MobileDataStateTracker(context, handler);
	    mNetTrackers[ConnectivityManager.TYPE_MOBILE] = mMobileDataStateTracker;
	
	/**
	   * Make the state of network connectivity conform to the preference settings.
	   * In this method, we only tear down a non-preferred network. Establishing
	   * a connection to the preferred network is taken care of when we handle
	   * the disconnect event from the non-preferred network
	   * (see {@link #handleDisconnect(NetworkInfo)}).
	   */
	  private void enforcePreference() {
	    if (mActiveNetwork == null)
	      return;
	
	    for (NetworkStateTracker t : mNetTrackers) {
	      if (t == mActiveNetwork) {
	        int netType = t.getNetworkInfo().getType();
	        int otherNetType = ((netType == ConnectivityManager.TYPE_WIFI) ?
	            ConnectivityManager.TYPE_MOBILE :
	            ConnectivityManager.TYPE_WIFI);

'701 Claims	Android Device with One or More Apps

	if (t.getNetworkInfo().getType() != mNetworkPreference) {
	NetworkStateTracker otherTracker = mNetTrackers[otherNetType];
	if (otherTracker.isAvailable()) {
	teardown(t);
	}
	}
	}
	}
	}
	
	/**
	* @see ConnectivityManager#getBackgroundDataSetting()
	*/
	public boolean getBackgroundDataSetting() {
	return Settings.Secure.getInt(mContext.getContentResolver(),
	Settings.Secure.BACKGROUND_DATA, 1) == 1;
	}
	
	/**
	* @see ConnectivityManager#setBackgroundDataSetting(boolean)
	*/
	public void setBackgroundDataSetting(boolean allowBackgroundDataUsage) {
	mContext.enforceCallingOrSelfPermission(
	android.Manifest.permission.CHANGE_BACKGROUND_DATA_SETTING,
	"ConnectivityService");
	
	if (getBackgroundDataSetting() == allowBackgroundDataUsage) return;
	
	Settings.Secure.putInt(mContext.getContentResolver(),

'701 Claims	Android Device with One or More Apps

	Settings.Secure.BACKGROUND_DATA, allowBackgroundDataUsage ? 1 : 0);
	
	Intent broadcast = new Intent(
	ConnectivityManager.ACTION_BACKGROUND_DATA_SETTING_CHANGED);
	mContext.sendBroadcast(broadcast);
	}
	
	/**
	* See if the other network is available to fail over to.
	* If is not available, we enable it anyway, so that it
	* will be able to connect when it does become available,
	* but we report a total loss of connectivity rather than
	* report that we are attempting to fail over.
	*/
	NetworkInfo switchTo = null;
	if (newNet.isAvailable()) {
	mActiveNetwork = newNet;
	switchTo = newNet.getNetworkInfo();
	switchTo.setFailover(true);
	if (!switchTo.isConnectedOrConnecting()) {
	newNet.reconnect();
	}
	} else {
	newNet.reconnect();
	}
	
	if (info.getType() == ConnectivityManager.TYPE_MOBILE) {
	otherNet = mWifiStateTracker;
	} else /* info().getType() == TYPE_WIFI */ {

’701 Claims	Android Device with One or More Apps

	otherNet = mMobileDataStateTracker;
	}
	
	int incrValue = ConnectivityManager.TYPE_MOBILE - ConnectivityManager.TYPE_WIFI;
	int stopValue = ConnectivityManager.TYPE_MOBILE + incrValue;
	
	SAMSUNG_PRIORART0005350, BatteryStats
	
	/**
	* A constant indicating a a wifi turn on timer
	*
	* {@hide}
	*/
	public static final int WIFI_TURNED_ON = 4;
	
	/**
	* A constant indicating a full wifi lock timer
	*
	* {@hide}
	*/
	public static final int FULL_WIFI_LOCK = 5;
	
	/**
	* A constant indicating a scan wifi lock timer
	*
	* {@hide}
	*/
	public static final int SCAN_WIFI_LOCK = 6;

'701 Claims	Android Device with One or More Apps
	/** * A constant indicating a wifi multicast timer * * {@hide} */ public static final int WIFI_MULTICAST_ENABLED = 7; /** * A constant indicating an audio turn on timer * * {@hide} */ public static final int AUDIO_TURNED_ON = 7; /** * A constant indicating a video turn on timer * * {@hide} */ public static final int VIDEO_TURNED_ON = 8; /** * Include all of the data in the stats, including previously saved data. */ public static final int STATS_TOTAL = 0; /** * Include only the last run in the stats. */ public static final int STATS_LAST = 1;

'701 Claims	Android Device with One or More Apps
	/** * Include only the current run in the stats. */ public static final int STATS_CURRENT = 2; /** * Include only the run since the last time the device was unplugged in the stats. */ public static final int STATS_UNPLUGGED = 3; public abstract void noteWifiTurnedOnLocked(); public abstract void noteWifiTurnedOffLocked(); public abstract void noteFullWifiLockAcquiredLocked(); public abstract void noteFullWifiLockReleasedLocked(); public abstract void noteScanWifiLockAcquiredLocked(); public abstract void noteScanWifiLockReleasedLocked(); public abstract void noteWifiMulticastEnabledLocked(); public abstract void noteWifiMulticastDisabledLocked(); /** * Returns the time in microseconds that the screen has been on while the device was * running on battery. * * {@hide} */ public abstract long getScreenOnTime(long batteryRealtime, int which); public static final int SCREEN_BRIGHTNESS_DARK = 0;

'701 Claims	Android Device with One or More Apps
	public static final int SCREEN_BRIGHTNESS_DIM = 1; public static final int SCREEN_BRIGHTNESS_MEDIUM = 2; public static final int SCREEN_BRIGHTNESS_LIGHT = 3; public static final int SCREEN_BRIGHTNESS_BRIGHT = 4; public static final int DATA_CONNECTION_NONE = 0; public static final int DATA_CONNECTION_GPRS = 1; public static final int DATA_CONNECTION_EDGE = 2; public static final int DATA_CONNECTION_UMTS = 3; public static final int DATA_CONNECTION_OTHER = 4; /** * Returns the time in microseconds that wifi has been on while the device was * running on battery. * * {@hide} */ public abstract long getWifiOnTime(long batteryRealtime, int which); /** * Returns the time in microseconds that bluetooth has been on while the device was * running on battery. * * {@hide} */ public abstract long getBluetoothOnTime(long batteryRealtime, int which); /** * Return whether we are currently running on battery. */

'701 Claims	Android Device with One or More Apps
	public abstract boolean getIsOnBattery(); /** * Returns the time that the radio was on for data transfers. * @return the uptime in microseconds while unplugged */ public abstract long getRadioDataUptime(); /** * Returns the current battery realtime in microseconds. * * @param curTime the amount of elapsed realtime in microseconds. */ public abstract long getBatteryRealtime(long curTime); /** * Returns the battery percentage level at the last time the device was unplugged from power, or * the last time it booted on battery power. */ public abstract int getDischargeStartLevel(); **Android 2.2** **GOOG-HEADWATER-00000029, SAMSUNG_PRIORART0005353, ConnectivityManager** /** * Class that answers queries about the state of network connectivity. It also

’701 Claims	Android Device with One or More Apps
	* notifies applications when network connectivity changes. Get an instance * of this class by calling * {@link android.content.Context#getSystemService(String) Context.getSystemService(Context.CONNECTIVITY_SERVICE)}. * <p> * The primary responsibilities of this class are to: * <ol> * <li>Monitor network connections (Wi-Fi, GPRS, UMTS, etc.)</li> * <li>Send broadcast intents when network connectivity changes</li> * <li>Attempt to "fail over" to another network when connectivity to a network * is lost</li> * <li>Provide an API that allows applications to query the coarse-grained or fine-grained * state of the available networks</li> * </ol> */ * A change in network connectivity has occurred. A connection has either * been established or lost. The NetworkInfo for the affected network is * sent as an extra; it should be consulted to see what kind of * connectivity event occurred. /** * Broadcast Action: The setting for **background data usage** has changed * values. Use {@link #getBackgroundDataSetting()} to get the current value. * <p> * If an application uses the network in the background, it should listen * for this broadcast and stop using the background data if the value is * false. */

<table>
<tr><th>'701 Claims</th><th>Android Device with One or More Apps</th></tr>
<tr><td></td><td><pre>
    @SdkConstant(SdkConstantType.BROADCAST_INTENT_ACTION)
    public static final String ACTION_BACKGROUND_DATA_SETTING_CHANGED =
            "android.net.conn.BACKGROUND_DATA_SETTING_CHANGED";

/**
     * The Default Mobile data connection. When active, all data traffic
     * will use this connection by default. Should not coexist with other
     * default connections.
     */
    public static final int TYPE_MOBILE   = 0;
    /**
     * The Default WIFI data connection. When active, all data traffic
     * will use this connection by default. Should not coexist with other
     * default connections.
     */
    public static final int TYPE_WIFI   = 1;

    /**
     * Returns the value of the setting for background data usage. If false,
     * applications should not use the network if the application is not in the
     * foreground. Developers should respect this setting, and check the value
     * of this before performing any background data operations.
     * <p>
     * All applications that have background services that use the network
     * should listen to {@link #ACTION_BACKGROUND_DATA_SETTING_CHANGED}.
     *
     * @return Whether background data usage is allowed.
</pre></td></tr>
</table>

'701 Claims	Android Device with One or More Apps

	*/
	  public boolean getBackgroundDataSetting() {
	      try {
	          return mService.getBackgroundDataSetting();
	      } catch (RemoteException e) {
	          // Err on the side of safety
	          return false;
	      }
	  }
	 /**
	   * Sets the value of the setting for background data usage.
	   *
	   * @param allowBackgroundData Whether an application should use data while
	   * it is in the background.
	   *
	   * @attr ref android.Manifest.permission#CHANGE_BACKGROUND_DATA_SETTING
	   * @see #getBackgroundDataSetting()
	   * @hide
	   */
	  public void setBackgroundDataSetting(boolean allowBackgroundData) {
	      try {
	          mService.setBackgroundDataSetting(allowBackgroundData);
	      } catch (RemoteException e) {
	      }
	  }
	
	  /**
	   * Sets the persisted value for enabling/disabling Mobile data.
	   *

'701 Claims	Android Device with One or More Apps

	* @param enabled Whether the mobile data connection should be
	   * used or not.
	   * @hide
	   */
	  public void setMobileDataEnabled(boolean enabled) {
	      try {
	          mService.setMobileDataEnabled(enabled);
	      } catch (RemoteException e) {
	      }
	  }
	
	SAMSUNG_PRIORART0005353, NetworkStateTracker
	
	  /**
	   * Record the roaming status of the device, and if it is a change from the previous
	   * status, send a notification to any listeners.
	   * @param isRoaming {@code true} if the device is now roaming, {@code false}
	   * if it is no longer roaming.
	   */
	  protected void setRoamingStatus(boolean isRoaming) {
	      if (isRoaming != mNetworkInfo.isRoaming()) {
	          mNetworkInfo.setRoaming(isRoaming);
	          Message msg = mTarget.obtainMessage(EVENT_ROAMING_CHANGED, mNetworkInfo);
	          msg.sendToTarget();
	      }
	  }
	
	  public static final int EVENT_ROAMING_CHANGED = 5;

'701 Claims	Android Device with One or More Apps

	SAMSUNG_PRIORART0005353, ThrottleManager
	/**
	 * Class that handles throttling. It provides read/write numbers per interface
	 * and methods to apply throttled rates.
	 * {@hide}
	 */
	
	    /**
	     * returns a long of the byte count either read or written on the named interface
	     * for the period described. Direction is either DIRECTION_RX or DIRECTION_TX and
	     * period may only be PERIOD_CYCLE for the current cycle (other periods may be supported
	     * in the future). Ago indicates the number of periods in the past to lookup - 0 means
	     * the current period, 1 is the last one, 2 was two periods ago..
	     * {@hide}
	     */
	    public long getByteCount(String iface, int direction, int period, int ago) {
	        try {
	            return mService.getByteCount(iface, direction, period, ago);
	        } catch (RemoteException e) {
	            return -1;
	        }
	    }
	
	    /**
	     * returns the number of bytes read+written after which a particular cliff
	     * takes effect on the named iface. Currently only cliff #1 is supported (1 step)
	     * {@hide}
	     */
	    public long getCliffThreshold(String iface, int cliff) {

’701 Claims	Android Device with One or More Apps

	try {
	return mService.getCliffThreshold(iface, cliff);
	} catch (RemoteException e) {
	return -1;
	}
	}
	SAMSUNG_PRIORART0005353, ConnectivityManagerMobileTest
	
	// help function to verify 3G connection
	public void verifyCellularConnection() {
	NetworkInfo extraNetInfo = cmActivity.mNetworkInfo;
	assertEquals("network type is not MOBILE", ConnectivityManager.TYPE_MOBILE,
	extraNetInfo.getType());
	assertTrue("not connected to cellular network", extraNetInfo.isConnected());
	assertTrue("no data connection", cmActivity.mState.equals(State.CONNECTED));
	}
	
	// Wait for the Wifi state to be DISABLED
	waitForWifiState(WifiManager.WIFI_STATE_DISABLED,
	STATE_TRANSITION_LONG_TIMEOUT);
	waitForNetworkState(ConnectivityManager.TYPE_WIFI, State.DISCONNECTED,
	STATE_TRANSITION_LONG_TIMEOUT);
	waitForNetworkState(ConnectivityManager.TYPE_MOBILE, State.CONNECTED,
	STATE_TRANSITION_LONG_TIMEOUT);
	
	//Prepare for connectivity state verification
	NetworkInfo networkInfo =
	cmActivity.mCM.getNetworkInfo(ConnectivityManager.TYPE_MOBILE);
	cmActivity.setStateTransitionCriteria(ConnectivityManager.TYPE_MOBILE,

'701 Claims	Android Device with One or More Apps

	networkInfo.getState(), NetworkState.DO_NOTHING,
	State.DISCONNECTED);
	networkInfo = cmActivity.mCM.getNetworkInfo(ConnectivityManager.TYPE_WIFI);
	cmActivity.setStateTransitionCriteria(ConnectivityManager.TYPE_WIFI, networkInfo.getState(),
	NetworkState.TO_CONNECTION, State.CONNECTED);
	
	// Wait for Wifi to be connected and mobile to be disconnected
	waitForNetworkState(ConnectivityManager.TYPE_WIFI, State.CONNECTED,
	STATE_TRANSITION_LONG_TIMEOUT);
	waitForNetworkState(ConnectivityManager.TYPE_MOBILE, State.DISCONNECTED,
	STATE_TRANSITION_LONG_TIMEOUT);
	
	// Test case 5: test connectivity from 3G to airplane mode, then to 3G again
	
	// Test case 6: test connectivity with airplane mode Wifi connected
	
	SAMSUNG_PRIORART0005353, ConnectivityService
	
	/*
	* Create the network state trackers for Wi-Fi and mobile
	* data. Maybe this could be done with a factory class,
	* but it's not clear that it's worth it, given that
	* the number of different network types is not going
	* to change very often.
	*/
	boolean noMobileData = !getMobileDataEnabled();
	for (int netType : mPriorityList) {
	switch (mNetAttributes[netType].mRadio) {

'701 Claims	Android Device with One or More Apps

	case ConnectivityManager.TYPE_WIFI:
	if (DBG) Slog.v(TAG, "Starting Wifi Service.");
	WifiStateTracker wst = new WifiStateTracker(context, mHandler);
	WifiService wifiService = new WifiService(context, wst);
	ServiceManager.addService(Context.WIFI_SERVICE, wifiService);
	wifiService.startWifi();
	mNetTrackers[ConnectivityManager.TYPE_WIFI] = wst;
	wst.startMonitoring();
	
	break;
	case ConnectivityManager.TYPE_MOBILE:
	mNetTrackers[netType] = new MobileDataStateTracker(context, mHandler,
	netType, mNetAttributes[netType].mName);
	mNetTrackers[netType].startMonitoring();
	if (noMobileData) {
	if (DBG) Slog.d(TAG, "tearing down Mobile networks due to setting");
	mNetTrackers[netType].teardown();
	}
	break;
	default:
	Slog.e(TAG, "Trying to create a DataStateTracker for an unknown radio type " +
	mNetAttributes[netType].mRadio);
	continue;
	}
	}
	
	/**
	* Sets the preferred network.
	* @param preference the new preference
	*/

'701 Claims	Android Device with One or More Apps

	public synchronized void setNetworkPreference(int preference) {
	enforceChangePermission();
	if (ConnectivityManager.isNetworkTypeValid(preference) &&
	mNetAttributes[preference] != null &&
	mNetAttributes[preference].isDefault()) {
	if (mNetworkPreference != preference) {
	persistNetworkPreference(preference);
	mNetworkPreference = preference;
	enforcePreference();
	}
	}
	}
	
	/**
	* Return NetworkInfo for the active (i.e., connected) network interface.
	* It is assumed that at most one network is active at a time. If more
	* than one is active, it is indeterminate which will be returned.
	* @return the info for the active network, or {@code null} if none is
	* active
	*/
	public NetworkInfo getActiveNetworkInfo() {
	enforceAccessPermission();
	for (int type=0; type <= ConnectivityManager.MAX_NETWORK_TYPE; type++) {
	if (mNetAttributes[type] == null || !mNetAttributes[type].isDefault()) {
	continue;
	}
	NetworkStateTracker t = mNetTrackers[type];
	NetworkInfo info = t.getNetworkInfo();
	if (info.isConnected()) {
	if (DBG && type != mActiveDefaultNetwork) Slog.e(TAG,

’701 Claims	Android Device with One or More Apps

	"connected default network is not " +
	"mActiveDefaultNetwork!");
	return info;
	}
	}
	return null;
	}
	
	// TODO - move this into the MobileDataStateTracker
	int usedNetworkType = networkType;
	if(networkType == ConnectivityManager.TYPE_MOBILE) {
	if (!getMobileDataEnabled()) {
	if (DBG) Slog.d(TAG, "requested special network with data disabled - rejected");
	return Phone.APN_TYPE_NOT_AVAILABLE;
	}
	if (TextUtils.equals(feature, Phone.FEATURE_ENABLE_MMS)) {
	usedNetworkType = ConnectivityManager.TYPE_MOBILE_MMS;
	} else if (TextUtils.equals(feature, Phone.FEATURE_ENABLE_SUPL)) {
	usedNetworkType = ConnectivityManager.TYPE_MOBILE_SUPL;
	} else if (TextUtils.equals(feature, Phone.FEATURE_ENABLE_DUN)) {
	usedNetworkType = ConnectivityManager.TYPE_MOBILE_DUN;
	} else if (TextUtils.equals(feature, Phone.FEATURE_ENABLE_HIPRI)) {
	usedNetworkType = ConnectivityManager.TYPE_MOBILE_HIPRI;
	}
	}
	
	int usedNetworkType = networkType;
	if (networkType == ConnectivityManager.TYPE_MOBILE) {
	if (TextUtils.equals(feature, Phone.FEATURE_ENABLE_MMS)) {

’701 Claims	Android Device with One or More Apps

```
                usedNetworkType = ConnectivityManager.TYPE_MOBILE_MMS;
            } else if (TextUtils.equals(feature, Phone.FEATURE_ENABLE_SUPL)) {
                usedNetworkType = ConnectivityManager.TYPE_MOBILE_SUPL;
            } else if (TextUtils.equals(feature, Phone.FEATURE_ENABLE_DUN)) {
                usedNetworkType = ConnectivityManager.TYPE_MOBILE_DUN;
            } else if (TextUtils.equals(feature, Phone.FEATURE_ENABLE_HIPRI)) {
                usedNetworkType = ConnectivityManager.TYPE_MOBILE_HIPRI;
            }
        }


    /**
     * @see ConnectivityManager#getBackgroundDataSetting()
     */
    public boolean getBackgroundDataSetting() {
        return Settings.Secure.getInt(mContext.getContentResolver(),
                Settings.Secure.BACKGROUND_DATA, 1) == 1;
    }

    /**
     * @see ConnectivityManager#setBackgroundDataSetting(boolean)
     */
    public void setBackgroundDataSetting(boolean allowBackgroundDataUsage) {
        mContext.enforceCallingOrSelfPermission(
                android.Manifest.permission.CHANGE_BACKGROUND_DATA_SETTING,
                "ConnectivityService");

        if (getBackgroundDataSetting() == allowBackgroundDataUsage) return;

        Settings.Secure.putInt(mContext.getContentResolver(),
```

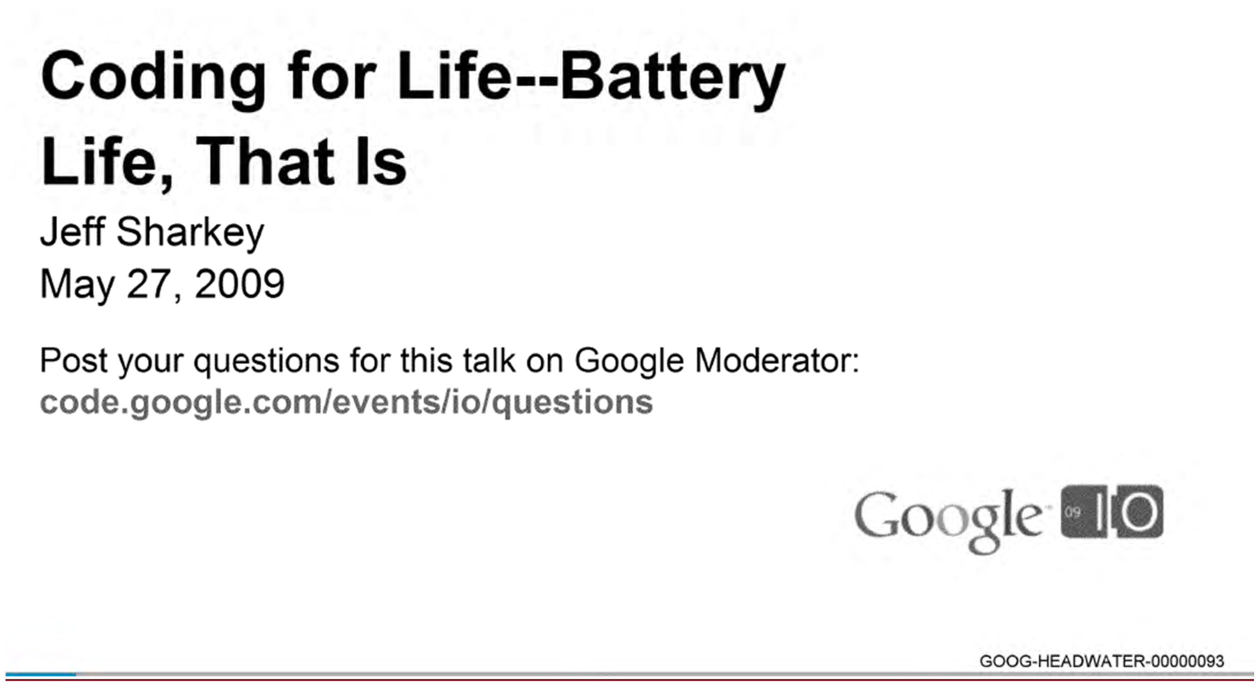
'701 Claims	Android Device with One or More Apps

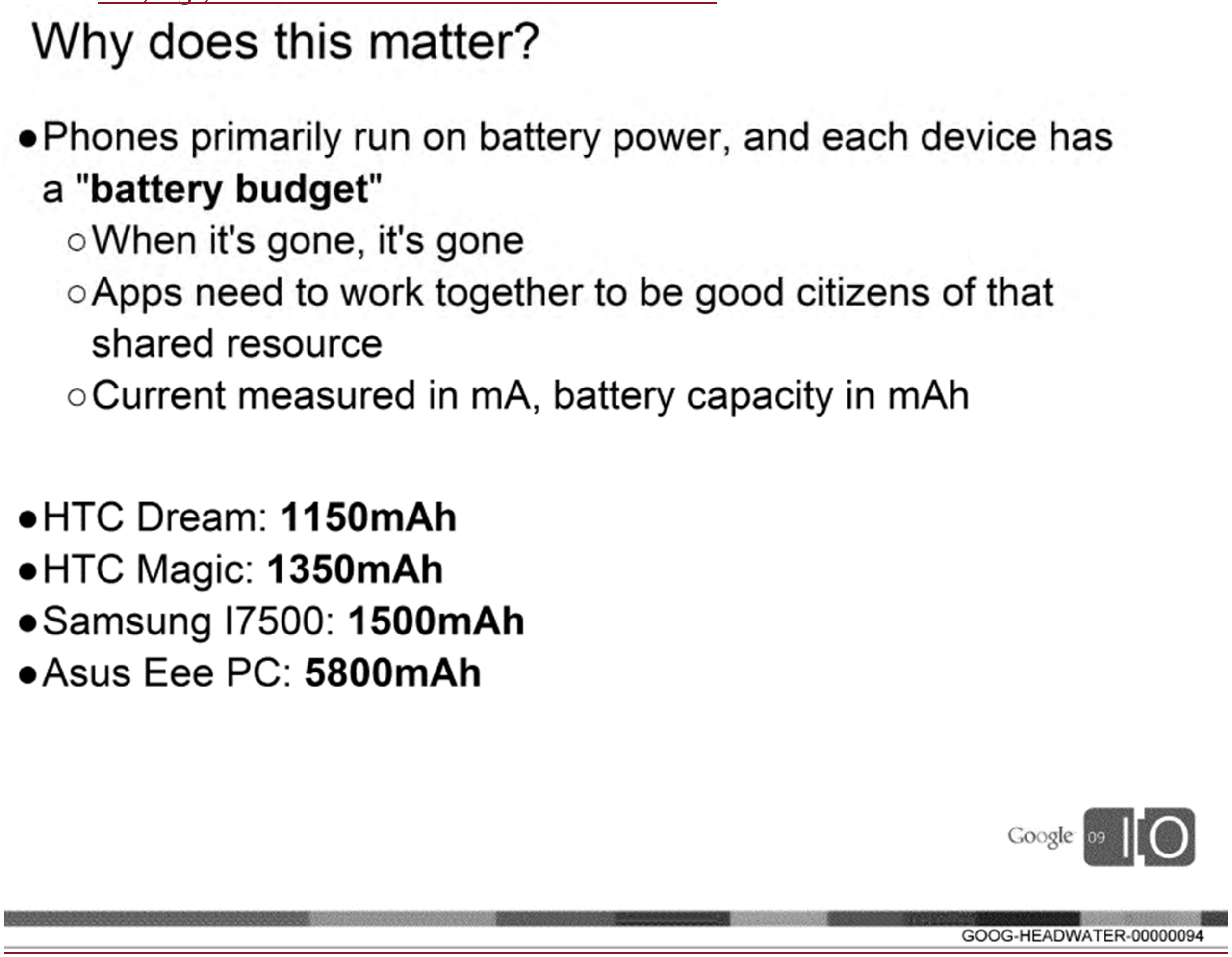
	Settings.Secure.BACKGROUND_DATA,
	allowBackgroundDataUsage ? 1 : 0);
	
	Intent broadcast = new Intent(
	ConnectivityManager.ACTION_BACKGROUND_DATA_SETTING_CHANGED);
	mContext.sendBroadcast(broadcast);
	}
	
	/**
	* @see ConnectivityManager#getMobileDataEnabled()
	*/
	public boolean getMobileDataEnabled() {
	enforceAccessPermission();
	boolean retVal = Settings.Secure.getInt(mContext.getContentResolver(),
	Settings.Secure.MOBILE_DATA, 1) == 1;
	if (DBG) Slog.d(TAG, "getMobileDataEnabled returning " + retVal);
	return retVal;
	}
	
	SAMSUNG_PRIORART0005353, BatteryStats
	
	/**
	* A class providing access to battery usage statistics, including information on
	* wakelocks, processes, packages, and services. All times are represented in microseconds
	* except where indicated otherwise.
	* @hide
	*/

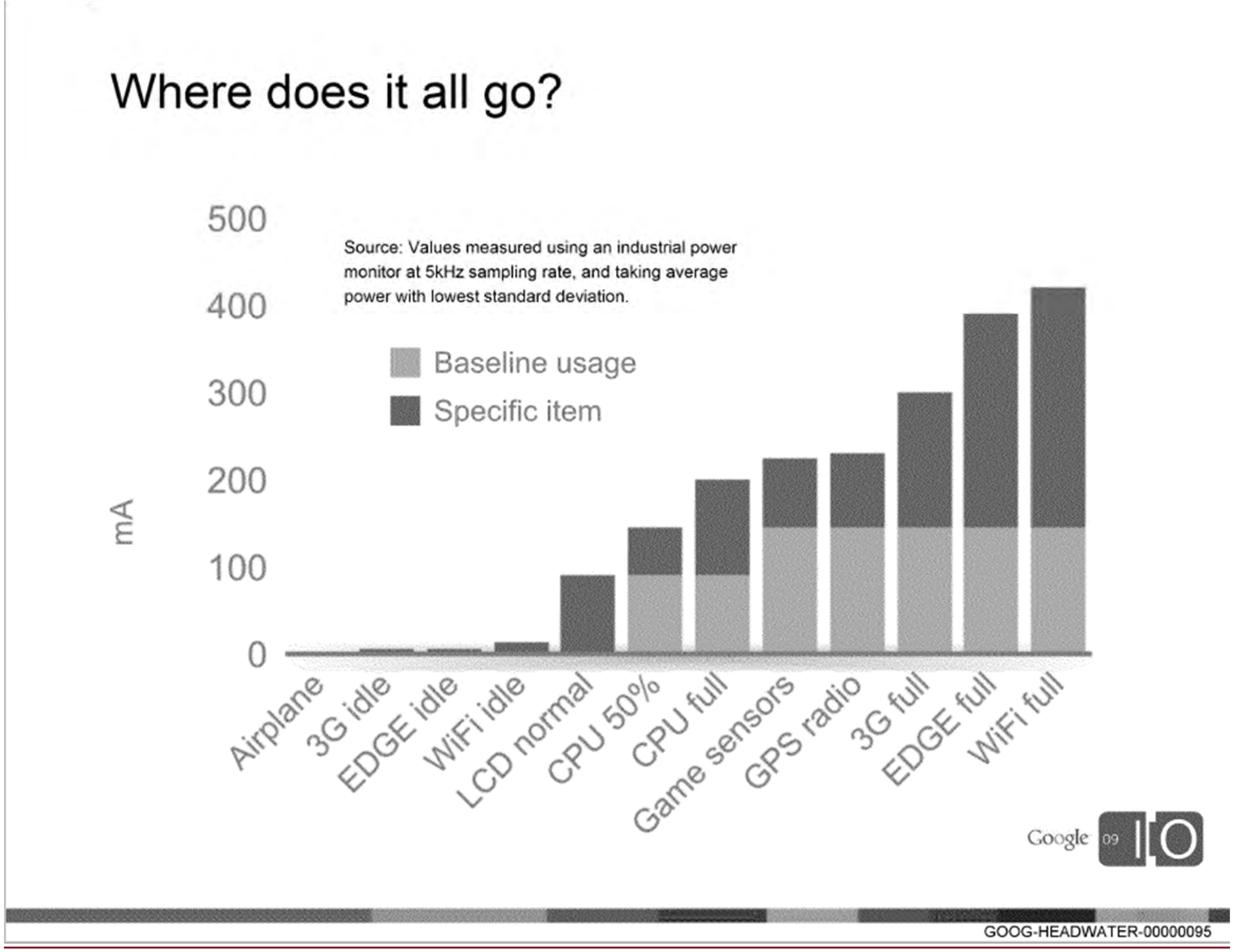
’701 Claims	Android Device with One or More Apps

	/**
	* A constant indicating a a wifi turn on timer
	*
	* {@hide}
	*/
	public static final int WIFI_TURNED_ON = 4;
	/**
	* A constant indicating an audio turn on timer
	*
	* {@hide}
	*/
	
	public static final int AUDIO_TURNED_ON = 7;
	
	/**
	* A constant indicating a video turn on timer
	*
	* {@hide}
	*/
	public static final int VIDEO_TURNED_ON = 8;
	
	public static final int SIGNAL_STRENGTH_NONE_OR_UNKNOWN = 0;
	public static final int SIGNAL_STRENGTH_POOR = 1;
	public static final int SIGNAL_STRENGTH_MODERATE = 2;
	public static final int SIGNAL_STRENGTH_GOOD = 3;
	public static final int SIGNAL_STRENGTH_GREAT = 4;
	
	static final String[] SIGNAL_STRENGTH_NAMES = {
	"none", "poor", "moderate", "good", "great"
	};

’701 Claims	Android Device with One or More Apps
	public static final int DATA_CONNECTION_NONE = 0; public static final int DATA_CONNECTION_GPRS = 1; public static final int DATA_CONNECTION_EDGE = 2; public static final int DATA_CONNECTION_UMTS = 3; public static final int DATA_CONNECTION_OTHER = 4; **GOOG-HEADWATER-00000092, Google I/O 2009 - Coding for Life -- Battery Life, That Is (June 2, 2009)** **Google I/O, 2009**

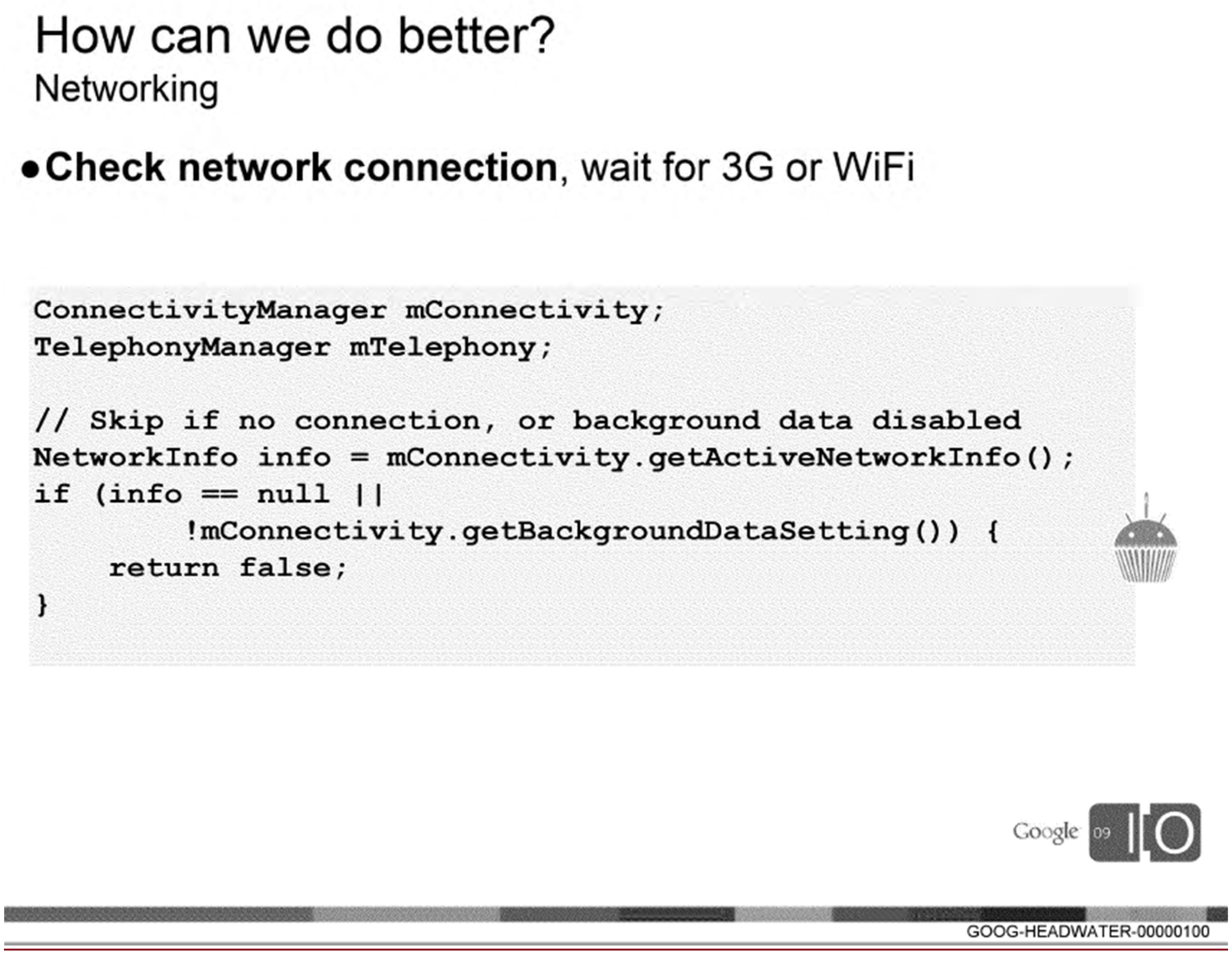
'701 Claims	Android Device with One or More Apps
	See, e.g., GOOG-HEADWATER-00000092 at 2:  **Coding for Life--Battery Life, That Is** Jeff Sharkey May 27, 2009 Post your questions for this talk on Google Moderator: **code.google.com/events/io/questions** Google 09 IO GOOG-HEADWATER-00000093

'701 Claims	Android Device with One or More Apps
	See, e.g., GOOG-HEADWATER-00000092 at 3:  Why does this matter? ● Phones primarily run on battery power, and each device has a "**battery budget**" ○ When it's gone, it's gone ○ Apps need to work together to be good citizens of that shared resource ○ Current measured in mA, battery capacity in mAh ● HTC Dream: **1150mAh** ● HTC Magic: **1350mAh** ● Samsung I7500: **1500mAh** ● Asus Eee PC: **5800mAh** Google 09 I/O GOOG-HEADWATER-00000094

'701 Claims	Android Device with One or More Apps
	See, e.g., GOOG-HEADWATER-00000092 at 4:  Where does it all go? Source: Values measured using an industrial power monitor at 5kHz sampling rate, and taking average power with lowest standard deviation. GOOG-HEADWATER-00000095

<table>
<tr><th>'701 Claims</th><th>Android Device with One or More Apps</th></tr>
<tr><td></td><td>

See, e.g., GOOG-HEADWATER-00000092 at 9:



How can we do better?

Networking

- **Check network connection**, wait for 3G or WiFi

<pre>
ConnectivityManager mConnectivity;
TelephonyManager mTelephony;

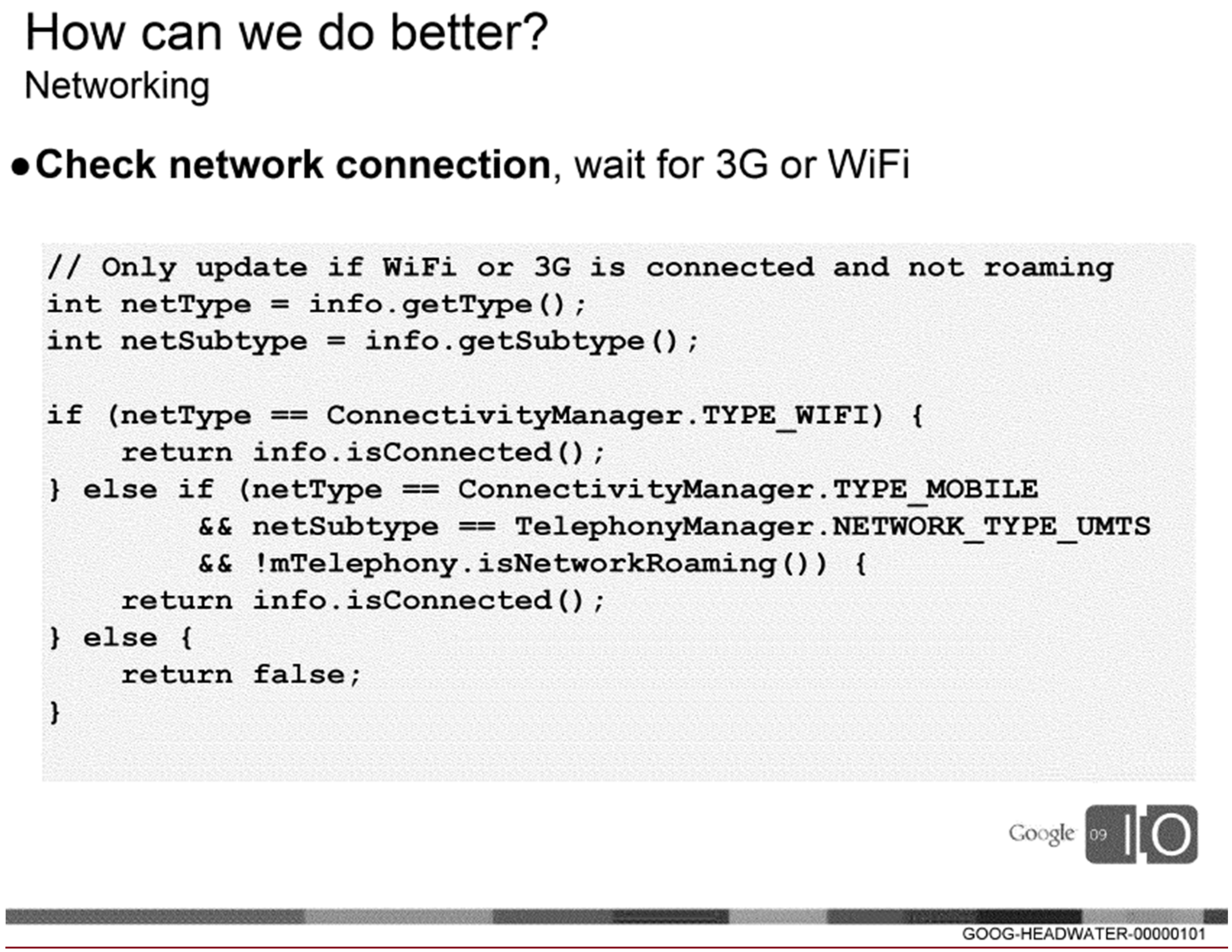
// Skip if no connection, or background data disabled
NetworkInfo info = mConnectivity.getActiveNetworkInfo();
if (info == null ||
        !mConnectivity.getBackgroundDataSetting()) {
    return false;
}
</pre>

Google 09 IO

GOOG-HEADWATER-00000100

See, e.g., GOOG-HEADWATER-00000092 at 11:

</td></tr>
</table>

'701 Claims	Android Device with One or More Apps
	

How can we do better?

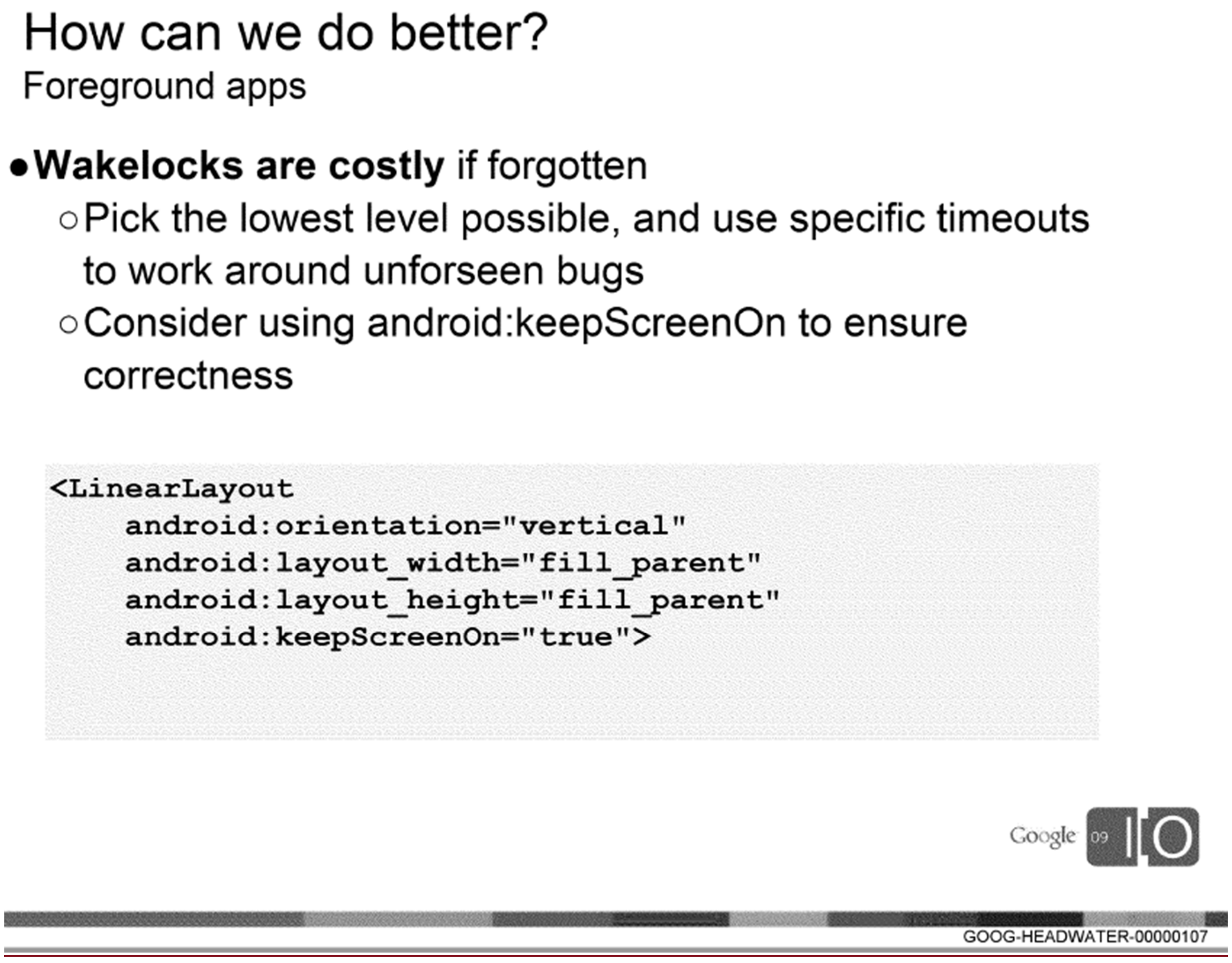
Networking

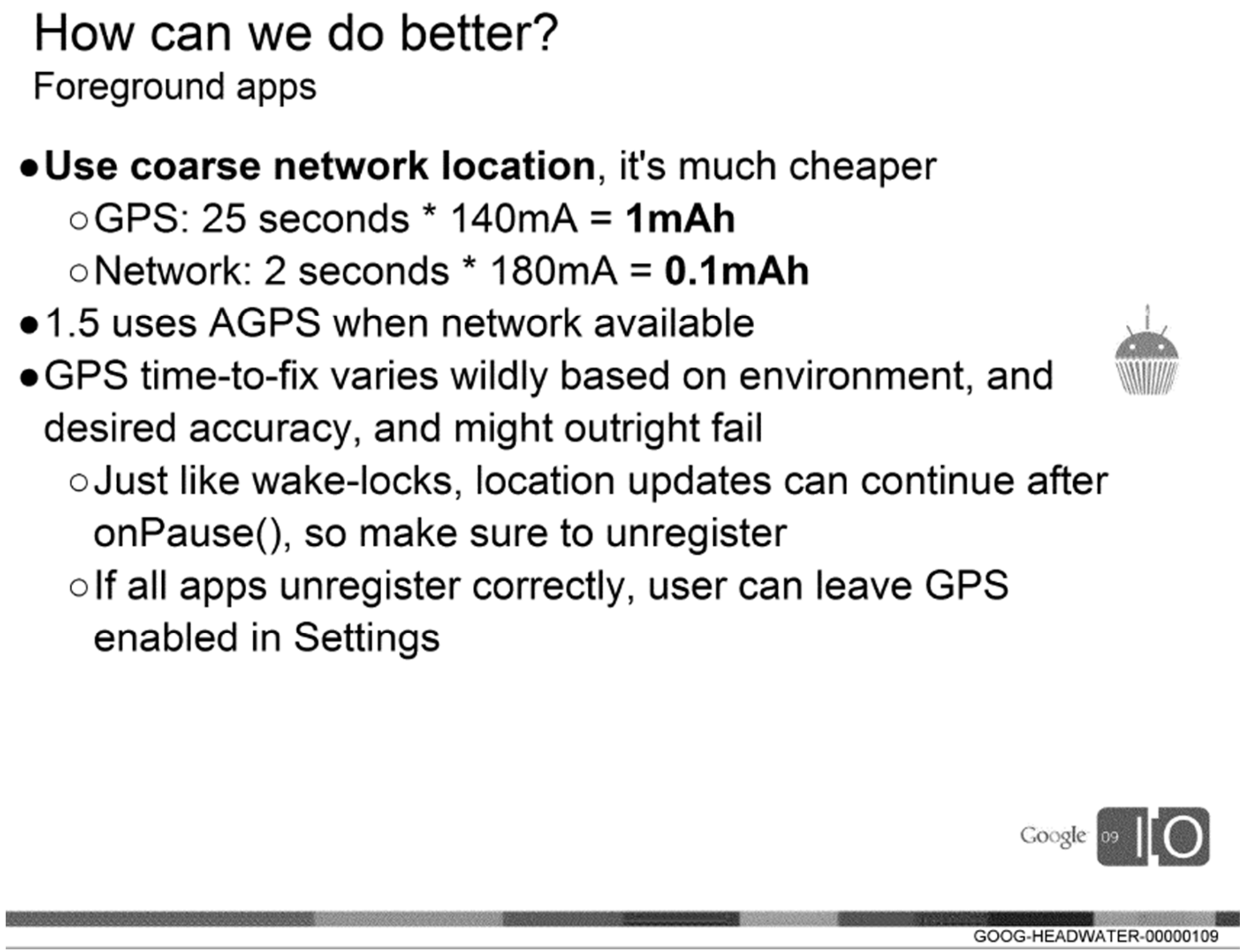
- **Check network connection**, wait for 3G or WiFi

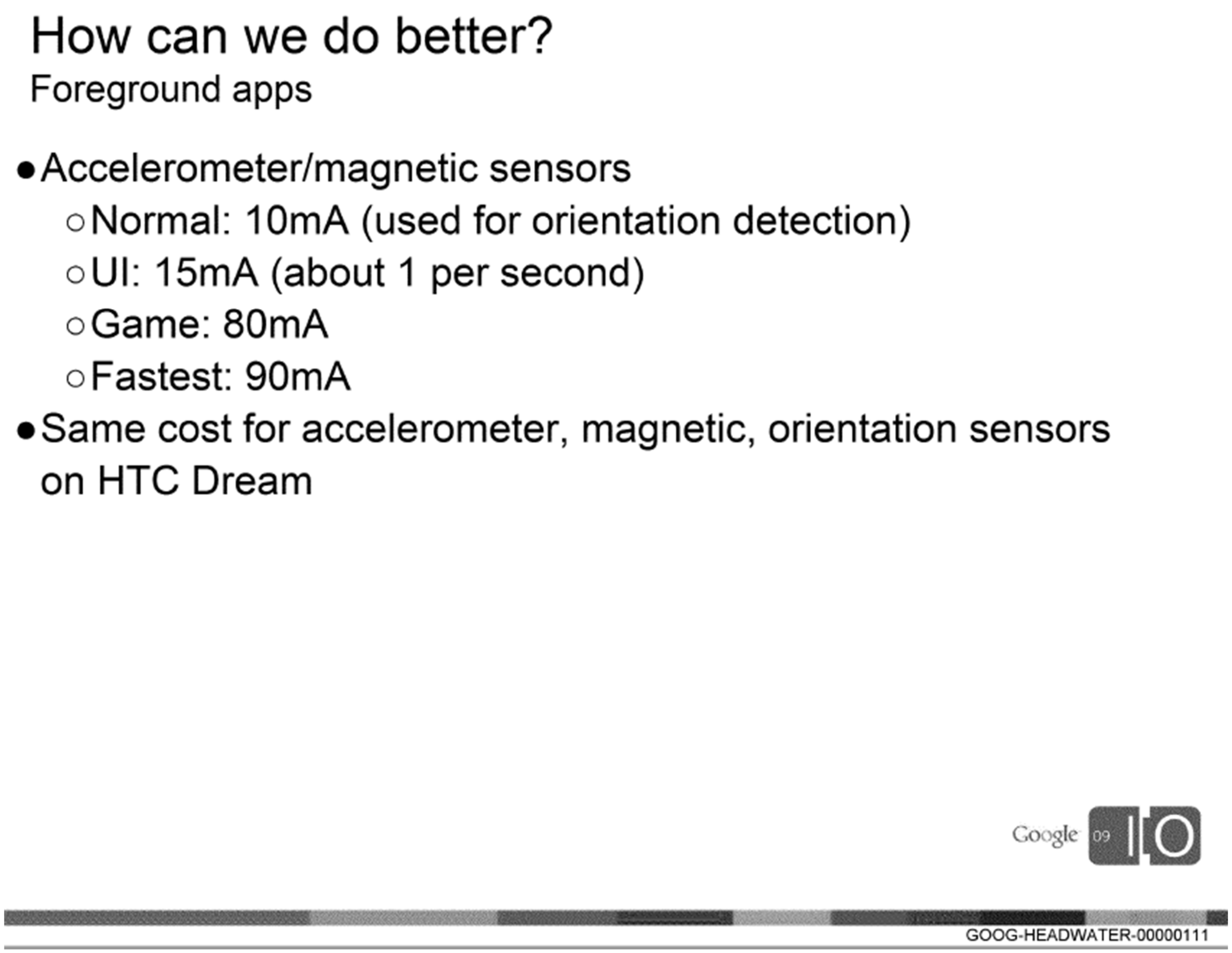
```
// Only update if WiFi or 3G is connected and not roaming
int netType = info.getType();
int netSubtype = info.getSubtype();

if (netType == ConnectivityManager.TYPE_WIFI) {
    return info.isConnected();
} else if (netType == ConnectivityManager.TYPE_MOBILE
        && netSubtype == TelephonyManager.NETWORK_TYPE_UMTS
        && !mTelephony.isNetworkRoaming()) {
    return info.isConnected();
} else {
    return false;
}
```

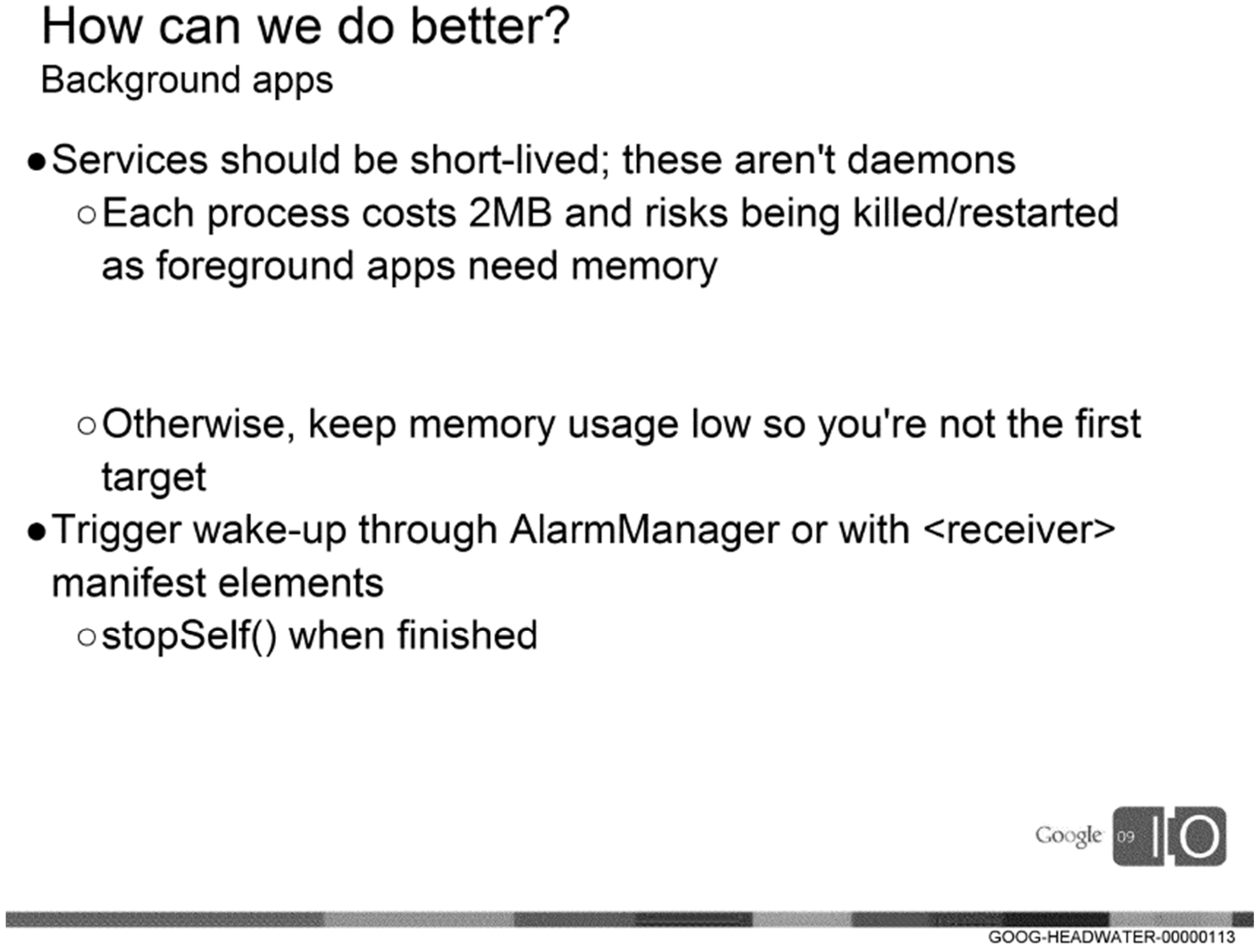
See, e.g., GOOG-HEADWATER-00000092 at 16:

'701 Claims	Android Device with One or More Apps

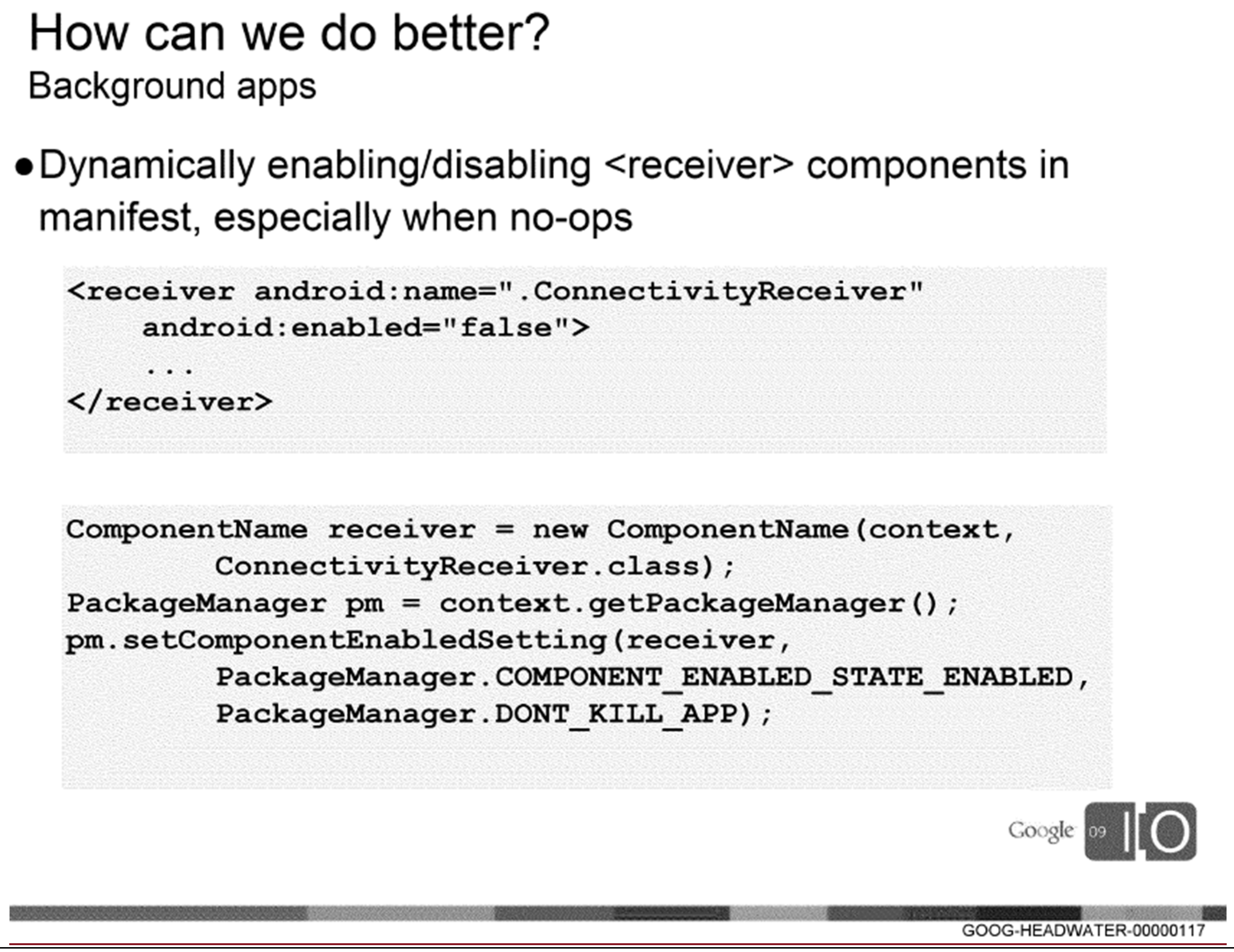
	
	How can we do better?
	Foreground apps
	● **Wakelocks are costly** if forgotten
	○ Pick the lowest level possible, and use specific timeouts to work around unforseen bugs
	○ Consider using android:keepScreenOn to ensure correctness
	`<LinearLayout`
	`    android:orientation="vertical"`
	`    android:layout_width="fill_parent"`
	`    android:layout_height="fill_parent"`
	`    android:keepScreenOn="true">`
	
	GOOG-HEADWATER-00000107
	See, e.g., GOOG-HEADWATER-00000092 at 18:

’701 Claims	Android Device with One or More Apps
	 How can we do better? Foreground apps ● **Use coarse network location**, it's much cheaper ○ GPS: 25 seconds * 140mA = **1mAh** ○ Network: 2 seconds * 180mA = **0.1mAh** ● 1.5 uses AGPS when network available ● GPS time-to-fix varies wildly based on environment, and desired accuracy, and might outright fail ○ Just like wake-locks, location updates can continue after onPause(), so make sure to unregister ○ If all apps unregister correctly, user can leave GPS enabled in Settings GOOG-HEADWATER-00000109 *See, e.g.*, GOOG-HEADWATER-00000092 at 20:

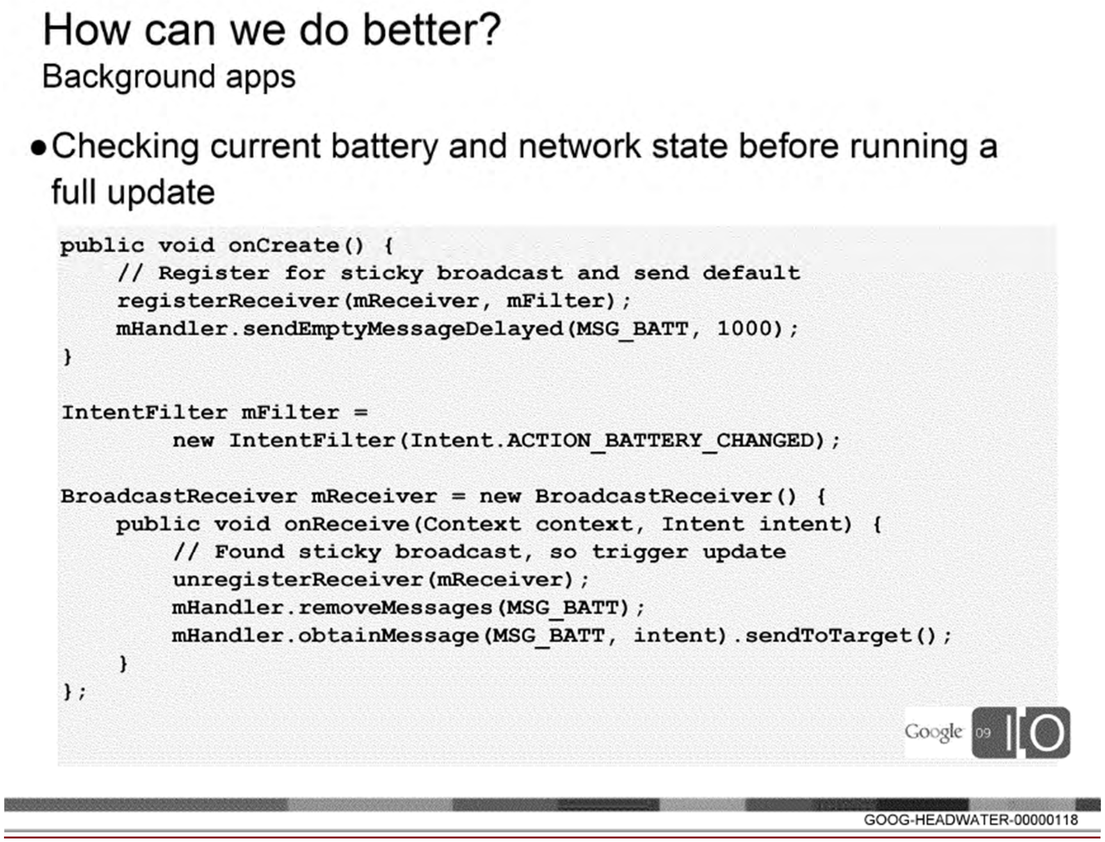
'701 Claims	Android Device with One or More Apps
	 How can we do better? Foreground apps ● Accelerometer/magnetic sensors ○ Normal: 10mA (used for orientation detection) ○ UI: 15mA (about 1 per second) ○ Game: 80mA ○ Fastest: 90mA ● Same cost for accelerometer, magnetic, orientation sensors on HTC Dream Google 09 IO GOOG-HEADWATER-00000111 *See, e.g.*, GOOG-HEADWATER-00000092 at 22:

'701 Claims	Android Device with One or More Apps
	 How can we do better? Background apps ● Services should be short-lived; these aren't daemons ○ Each process costs 2MB and risks being killed/restarted as foreground apps need memory ○ Otherwise, keep memory usage low so you're not the first target ● Trigger wake-up through AlarmManager or with <receiver> manifest elements ○ stopSelf() when finished Google 09 IO GOOG-HEADWATER-00000113 *See, e.g.*, GOOG-HEADWATER-00000092 at 26:

’701 Claims	Android Device with One or More Apps

	
	How can we do better?
	Background apps
	● Dynamically enabling/disabling <receiver> components in manifest, especially when no-ops
	`<receiver android:name=".ConnectivityReceiver"`
	`    android:enabled="false">`
	`    ...`
	`</receiver>`
	`ComponentName receiver = new ComponentName(context,`
	`        ConnectivityReceiver.class);`
	`PackageManager pm = context.getPackageManager();`
	`pm.setComponentEnabledSetting(receiver,`
	`        PackageManager.COMPONENT_ENABLED_STATE_ENABLED,`
	`        PackageManager.DONT_KILL_APP);`
	Google 09 IO
	GOOG-HEADWATER-00000117

<table>
<tr><th>'701 Claims</th><th>Android Device with One or More Apps</th></tr>
<tr><td></td><td>
<em>See, e.g.</em>, GOOG-HEADWATER-00000092 at 27:



How can we do better?

Background apps

● Checking current battery and network state before running a full update

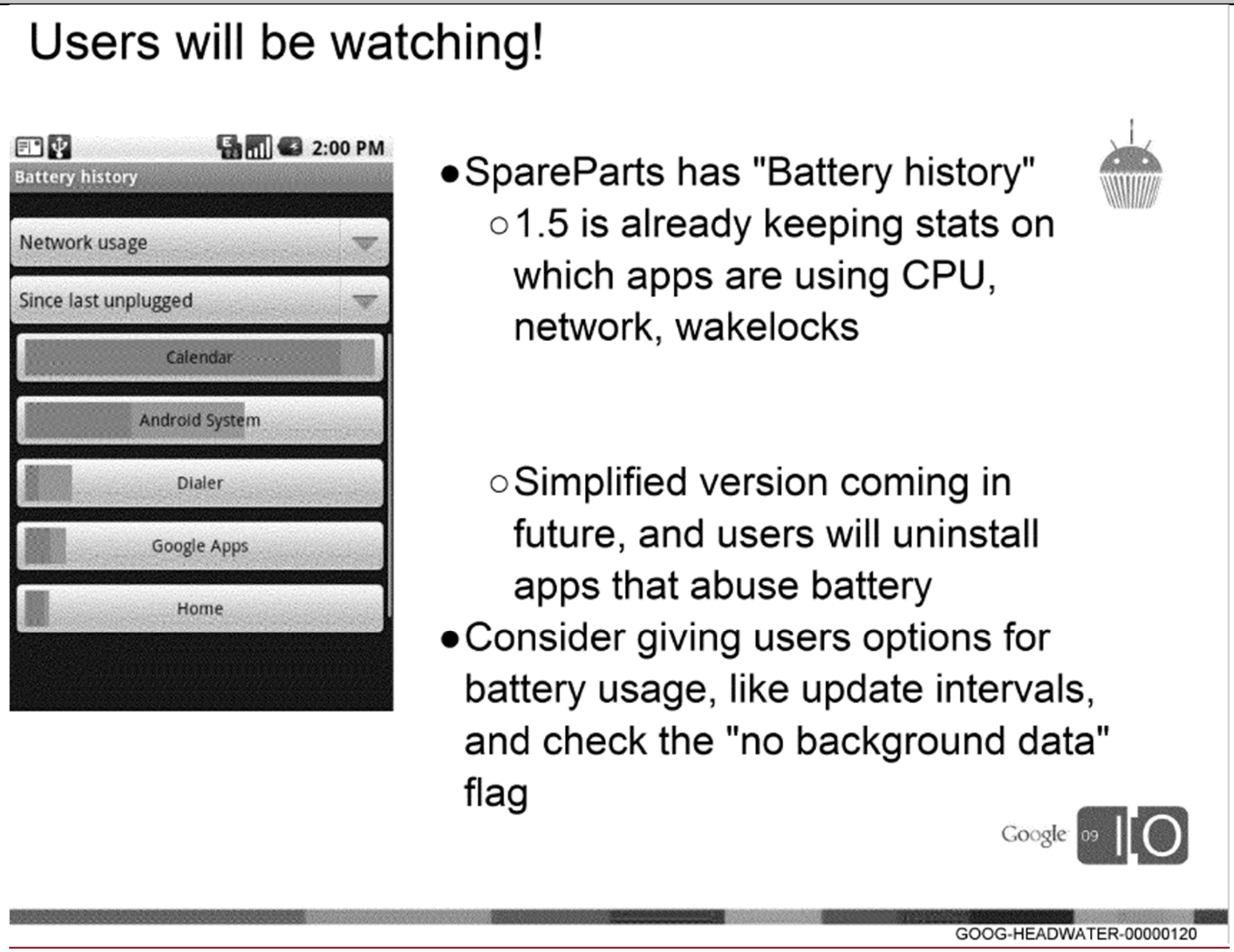
<pre>
public void onCreate() {
    // Register for sticky broadcast and send default
    registerReceiver(mReceiver, mFilter);
    mHandler.sendEmptyMessageDelayed(MSG_BATT, 1000);
}

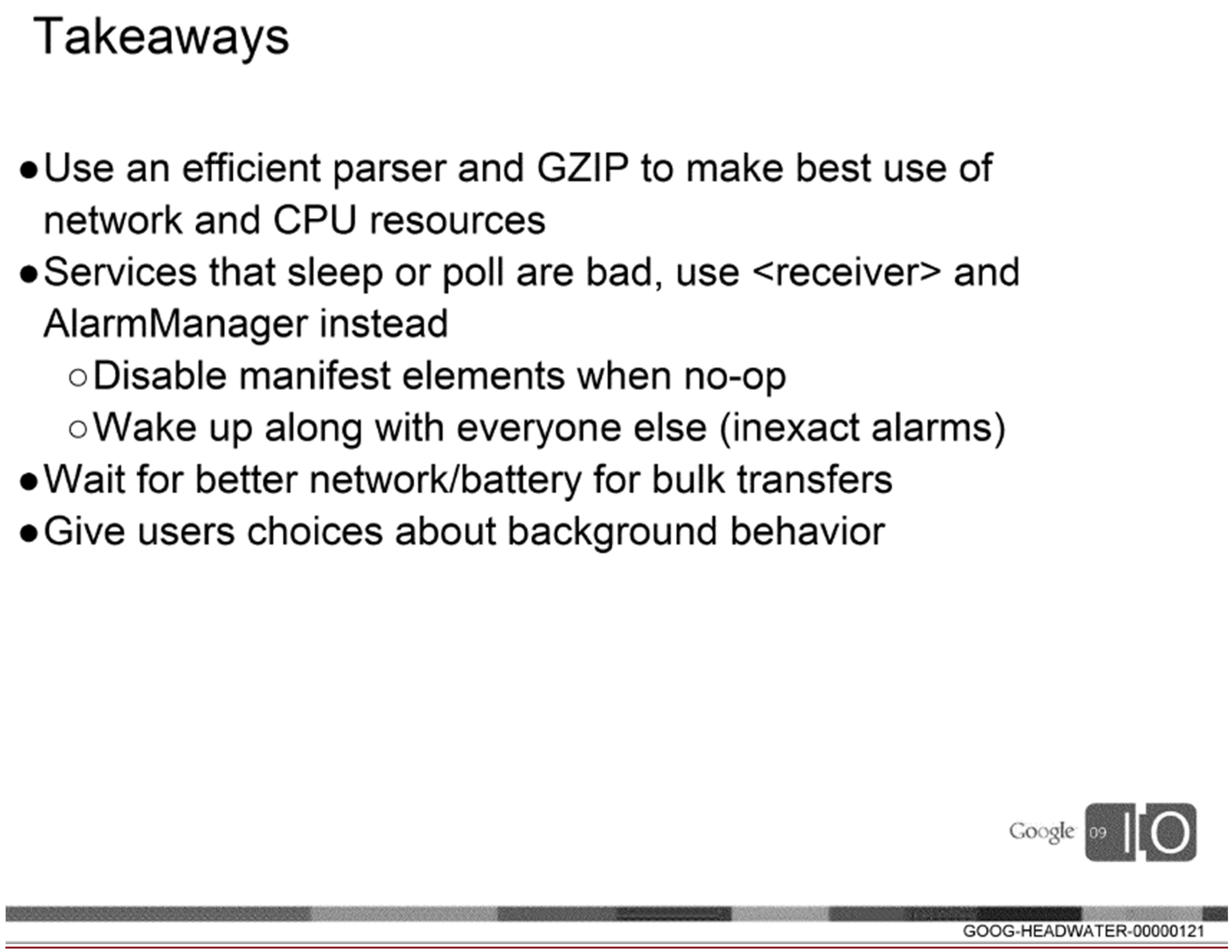
IntentFilter mFilter =
        new IntentFilter(Intent.ACTION_BATTERY_CHANGED);

BroadcastReceiver mReceiver = new BroadcastReceiver() {
    public void onReceive(Context context, Intent intent) {
        // Found sticky broadcast, so trigger update
        unregisterReceiver(mReceiver);
        mHandler.removeMessages(MSG_BATT);
        mHandler.obtainMessage(MSG_BATT, intent).sendToTarget();
    }
};
</pre>

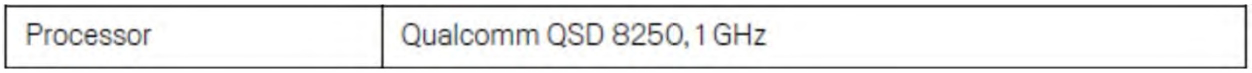
GOOG-HEADWATER-00000118

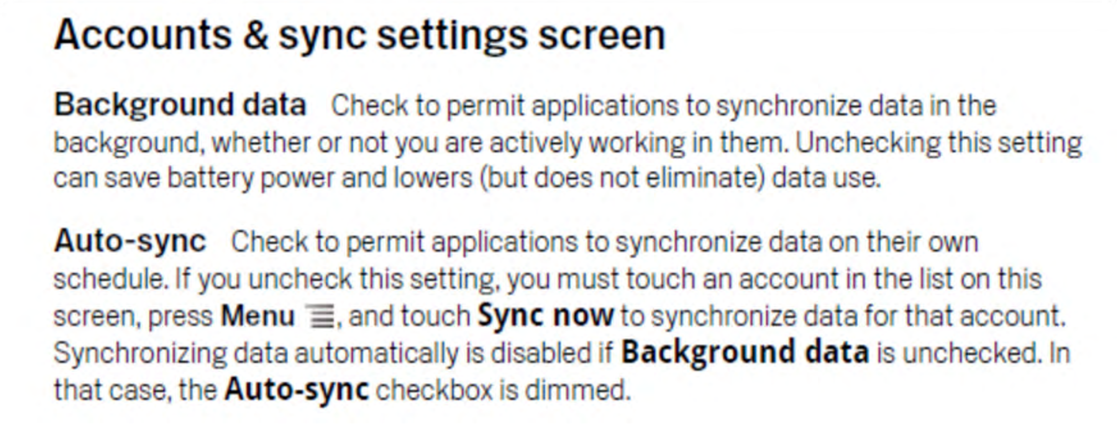
<em>See, e.g.</em>, GOOG-HEADWATER-00000092 at 29:
</td></tr>
</table>

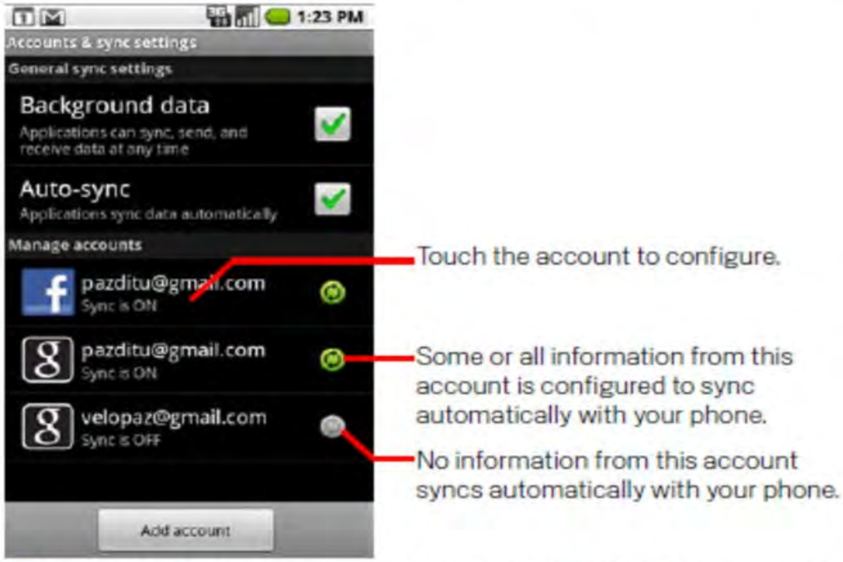
'701 Claims	Android Device with One or More Apps
	 2:00 PM Battery history Network usage Since last unplugged Calendar Android System Dialer Google Apps Home Google 09 IO GOOG-HEADWATER-00000120 Users will be watching! ● SpareParts has "Battery history" ○ 1.5 is already keeping stats on which apps are using CPU, network, wakelocks ○ Simplified version coming in future, and users will uninstall apps that abuse battery ● Consider giving users options for battery usage, like update intervals, and check the "no background data" flag *See, e.g.*, GOOG-HEADWATER-00000092 at 30:

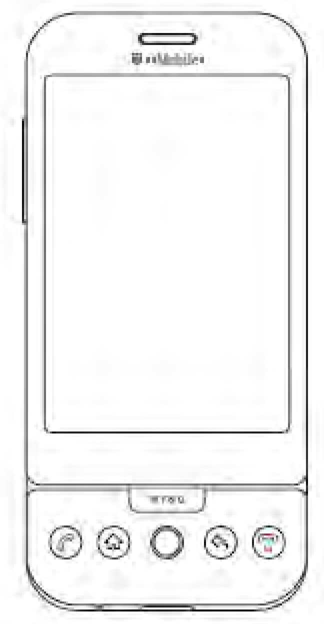
’701 Claims	Android Device with One or More Apps
	 Takeaways ● Use an efficient parser and GZIP to make best use of network and CPU resources ● Services that sleep or poll are bad, use <receiver> and AlarmManager instead ○ Disable manifest elements when no-op ○ Wake up along with everyone else (inexact alarms) ● Wait for better network/battery for bulk transfers ● Give users choices about background behavior Google 09 IO GOOG-HEADWATER-00000121 **GreenPower App**

'701 Claims	Android Device with One or More Apps
	POUZERATE0000196 (GreenPower User Guide) ("Manage Mobile Network If this setting is selected, then Green Power will regularly turn on and off the Mobile Network connection, based on the durations specified in the settings below. If this setting is not selected, then Green Power will leave the Mobile Network as it is, never turning it on or off. Please note that in order for Green Power to turn on / off Mobile Network, this one has to be manually enabled by the user first in the phone settings (Wireless & networks → Mobile Network) or in Green Power settings (Global wireless settings → Mobile Network) . Green Power can't itself turn on Mobile Network as this is a limitation of the Android system for security and cost reasons.").
[1b] a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN;	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for claims [1 pre] – [1a]. **GreenPower App** POUZERATE0000196 (GreenPower User Guide) ("Manage Wifi If this setting is selected, then Green Power will regularly turn on and off the Wifi connection, based on the durations specified in the settings below. If this setting is not selected, then Green Power will leave the Wifi as it is, never turning it on or off.

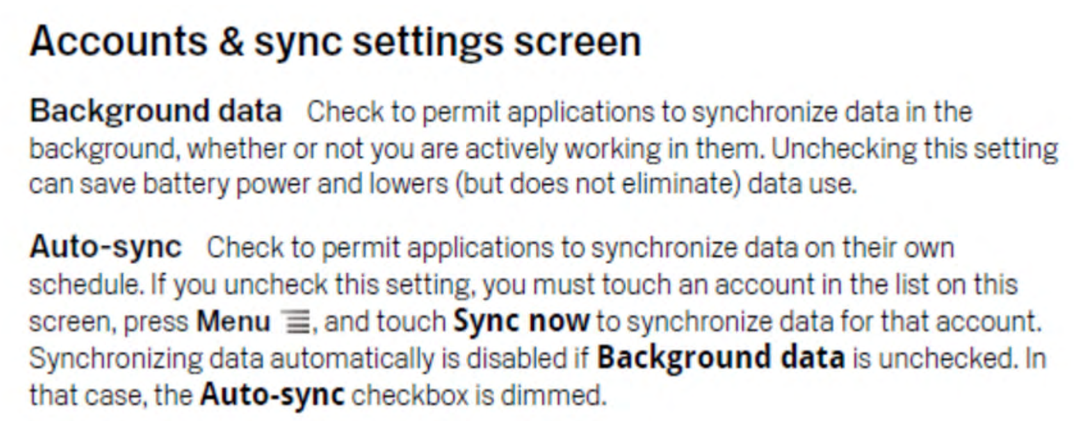
'701 Claims	Android Device with One or More Apps
	Please note that if you manually switches off the Wifi, then Green Power will unselect the "Manage Wifi" setting in order not to automatically switch on the Wifi again despite your manual action. Then, if you switch back on the Wifi or reselect "Manage Wifi" setting, Green Power will resume managing Wifi connection.").
[1c] one or more processors configured to	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for claim [1pre]. In addition: *See, e.g.*, SAMSUNG_PRIORART0000001 (Nexus) at 331:  Processor | Qualcomm QSD 8250, 1 GHz *See, e.g.*, SAMSUNG_PRIORART0000001 (Nexus) at 320:

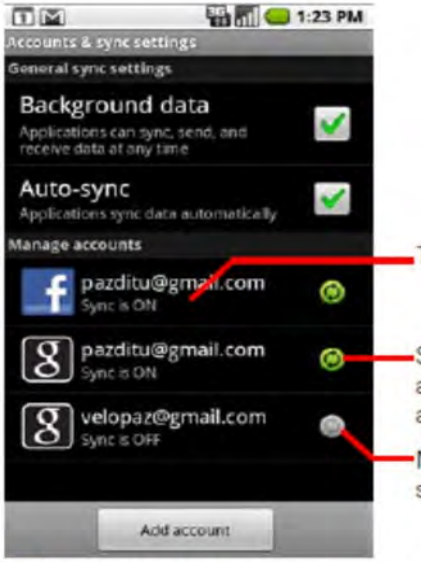
'701 Claims	Android Device with One or More Apps
	 **Accounts & sync settings screen** **Background data** Check to permit applications to synchronize data in the background, whether or not you are actively working in them. Unchecking this setting can save battery power and lowers (but does not eliminate) data use. **Auto-sync** Check to permit applications to synchronize data on their own schedule. If you uncheck this setting, you must touch an account in the list on this screen, press **Menu** ☰, and touch **Sync now** to synchronize data for that account. Synchronizing data automatically is disabled if **Background data** is unchecked. In that case, the **Auto-sync** checkbox is dimmed. SAMSUNG_PRIORART0000001 (Nexus) at 115-116 ("You can configure background data use and synchronization options for all of the applications on your phone. You can also configure what kinds of data you synchronize for each account. Some applications, such as Gmail and Calendar, have their own synchronization settings."). SAMSUNG_PRIORART0000001 (Nexus) at 115-116:

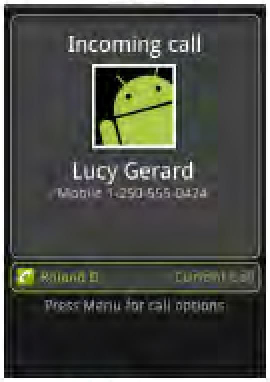
'701 Claims	Android Device with One or More Apps
	The screen displays your current sync settings and a list of your current accounts.  Accounts & sync settings General sync settings Background data Applications can sync, send, and receive data at any time ☑ Auto-sync Applications sync data automatically ☑ Manage accounts pazditu@gmail.com Sync is ON pazditu@gmail.com Sync is ON velopaz@gmail.com Sync is OFF Add account Touch the account to configure. Some or all information from this account is configured to sync automatically with your phone. No information from this account syncs automatically with your phone. 🟢 indicates that some or all of an account's information is configured to sync automatically with your phone. ⚪ indicates that none of an account's information is configured to sync automatically with your phone. **HTC Dream / T-Mobile G1** The HTC Dream / T-Mobile G1 is an example of an Android smartphone. SAMSUNG_PRIORART0005184

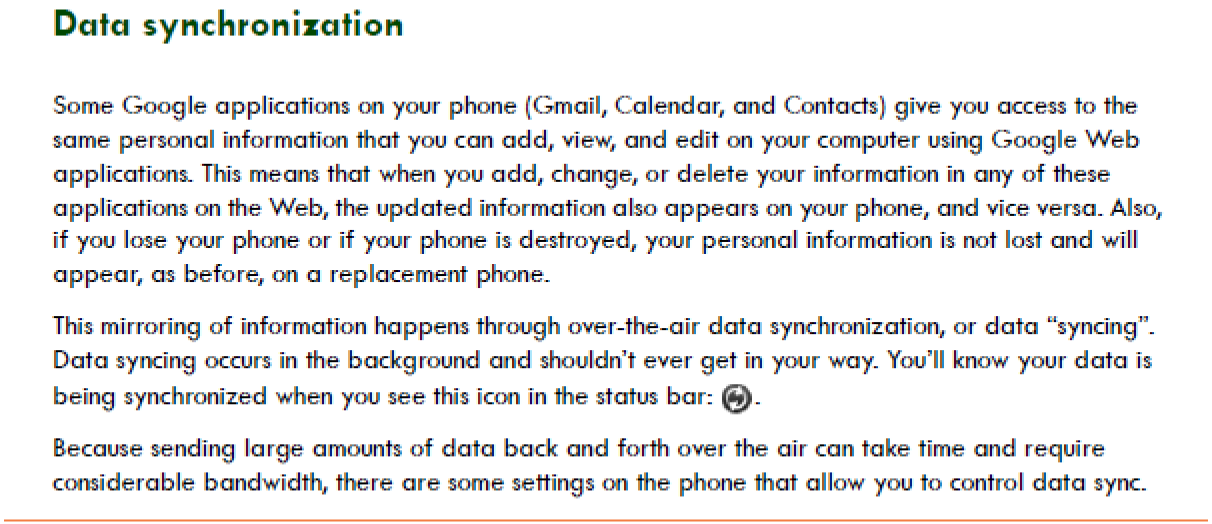
’701 Claims	Android Device with One or More Apps
	 T-Mobile G1 SAMSUNG_PRIORART0005177 Android™ mobile technology platform R1.0 Document Rev 08 - September 8, 2008 Copyright 2008 © Google, Inc. All rights reserved. **Samsung GT-I7500 Galaxy**

'701 Claims	Android Device with One or More Apps
	The Samsung GT-I7500 Galaxy is an example of an Android smartphone. SAMSUNG_PRIORART0005494 at pgs. PDF 2, 19: SAMSUNG OK Earpiece Volume key Confirm key Menu key Dial key Proximity sensor Touch screen 4-way navigation key Back key Home key Power key Mouthpiece
[1d] determine, for a first end-user application capable of running in a background state and capable of	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: **Nexus One**

’701 Claims	Android Device with One or More Apps
running as a foreground application, whether the application is running in a background state or as a foreground application, and control, via an application program interface (API), application access for Internet service activities provided through the WWAN modem and the WLAN modem, to, based on a first differential traffic control policy, selectively block and allow access by the first end-user application to the WWAN modem at a time when data for Internet service activities is communicated through a WWAN modem connection to the at least one WWAN,	*See, e.g.*, SAMSUNG_PRIORART0000001 (Nexus) at 320:  **Accounts & sync settings screen** **Background data** Check to permit applications to synchronize data in the background, whether or not you are actively working in them. Unchecking this setting can save battery power and lowers (but does not eliminate) data use. **Auto-sync** Check to permit applications to synchronize data on their own schedule. If you uncheck this setting, you must touch an account in the list on this screen, press **Menu** ☰, and touch **Sync now** to synchronize data for that account. Synchronizing data automatically is disabled if **Background data** is unchecked. In that case, the **Auto-sync** checkbox is dimmed. SAMSUNG_PRIORART0000001 (Nexus) at 115-116 (“You can configure background data use and synchronization options for all of the applications on your phone. You can also configure what kinds of data you synchronize for each account. Some applications, such as Gmail and Calendar, have their own synchronization settings.”). SAMSUNG_PRIORART0000001 (Nexus) at 115-116:

'701 Claims	Android Device with One or More Apps
	The screen displays your current sync settings and a list of your current accounts.  Accounts & sync settings General sync settings Background data Applications can sync, send, and receive data at any time ☑ Auto-sync Applications sync data automatically ☑ Manage accounts pazditu@gmail.com Sync is ON pazditu@gmail.com Sync is ON velopaz@gmail.com Sync is OFF Add account Touch the account to configure. Some or all information from this account is configured to sync automatically with your phone. No information from this account syncs automatically with your phone. ● indicates that some or all of an account's information is configured to sync automatically with your phone. ○ indicates that none of an account's information is configured to sync automatically with your phone. HTC Dream / T-Mobile G1 SAMSUNG_PRIORART0005260

'701 Claims	Android Device with One or More Apps
	Answer incoming, hold call in progress The incoming call will display in the foreground.  Incoming call Lucy Gerard Mobile 1-250-555-0424 Roland D Current call Press Menu for call options Press CALL to answer the incoming call and place the call in progress on hold. You can also press MENU and select Hold call in progress & answer. To switch calls at any time, press MENU and select Switch calls. SAMSUNG_PRIORART0005207-09

’701 Claims	Android Device with One or More Apps
	 **Data synchronization** Some Google applications on your phone (Gmail, Calendar, and Contacts) give you access to the same personal information that you can add, view, and edit on your computer using Google Web applications. This means that when you add, change, or delete your information in any of these applications on the Web, the updated information also appears on your phone, and vice versa. Also, if you lose your phone or if your phone is destroyed, your personal information is not lost and will appear, as before, on a replacement phone. This mirroring of information happens through over-the-air data synchronization, or data “syncing”. Data syncing occurs in the background and shouldn’t ever get in your way. You’ll know your data is being synchronized when you see this icon in the status bar: ⟳. Because sending large amounts of data back and forth over the air can take time and require considerable bandwidth, there are some settings on the phone that allow you to control data sync.

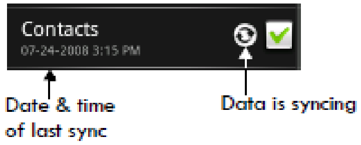
Sync by application

To control synchronization for Gmail, Calendar, and Contacts, do the following:

1 Press HOME, then press MENU and select Settings.

2 Select Data synchronization.

When any of the applications are synchronizing, you will see the "sync" icon: ⟳. You will also see the last time your data was synchronized.



3 By default, the personal information in Gmail, Calendar, and Contacts will sync whenever you make a change or receive a new message. You can change this behavior:

Auto-sync When selected, Auto-sync will sync Gmail, Calendar, and Contacts automatically, as you make changes or receive new messages. When OFF, information will not be synced automatically, although you can force a sync by using the individual application check boxes described below.

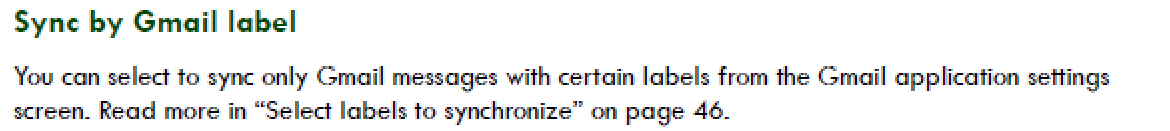
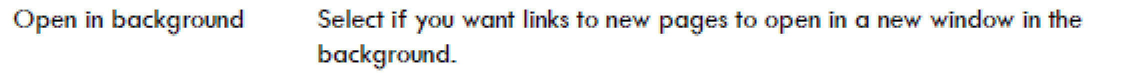
Gmail Clear this check box to exclude Gmail from auto-sync. To force a sync, either clear the check box then select it, or press MENU and select Sync now. To control sync by Gmail label, you must do so from the Gmail settings screen. Read more in "Select labels to synchronize" on page 46.

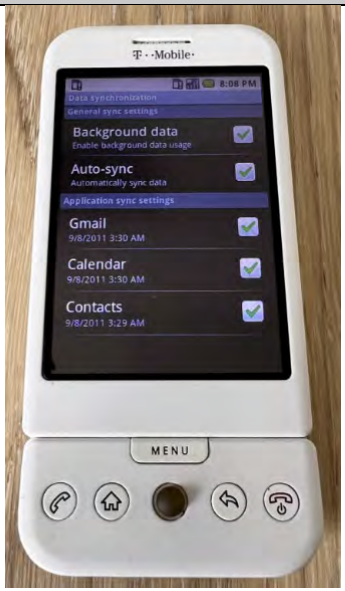
Calendar Clear this check box to exclude Calendar from auto-sync. To force a sync, either clear the check box then select it, or press MENU and select Sync now.

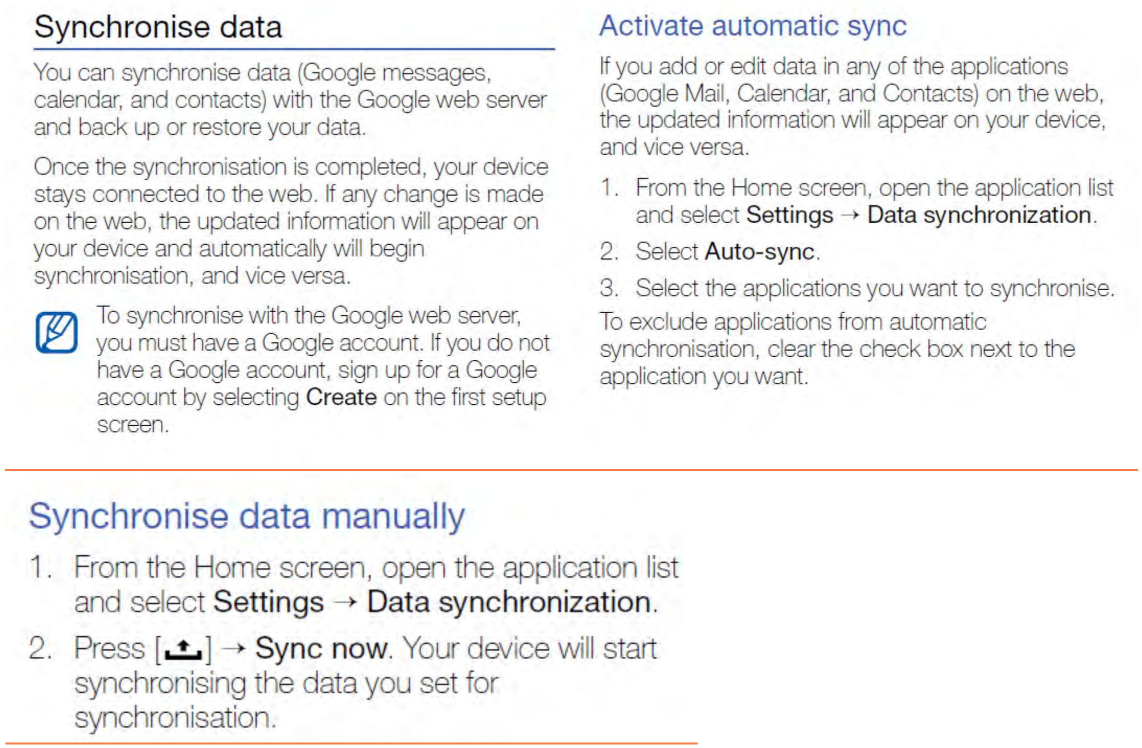
Contacts Clear this check box to exclude Contacts from auto-sync. To force a sync, either clear the check box then select it, or press MENU and select Sync now.

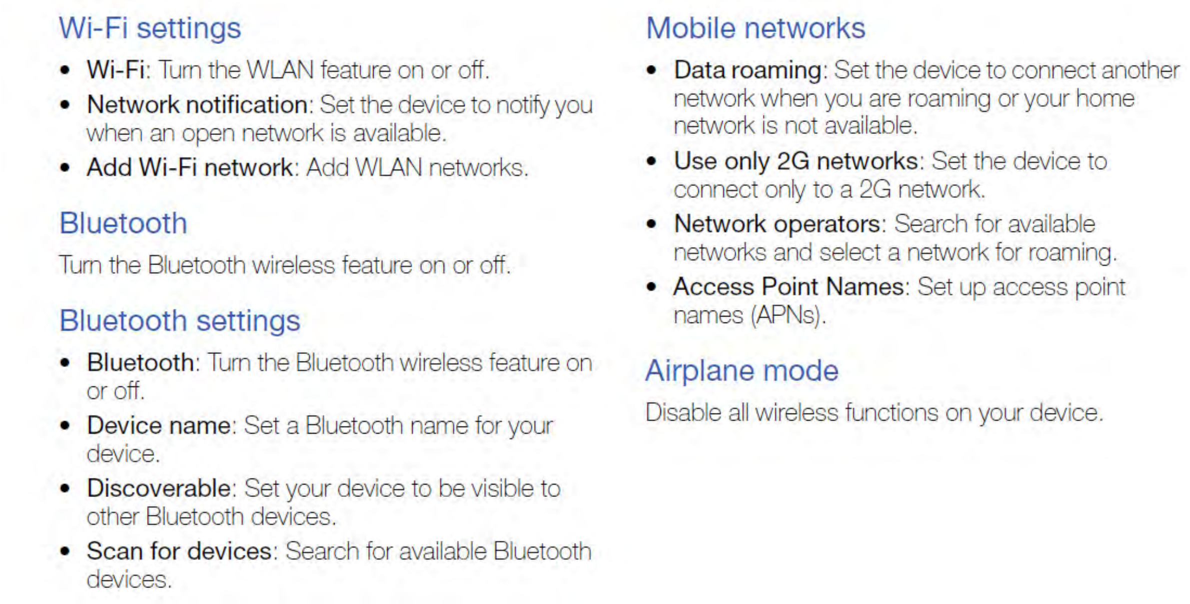
Cancel sync During a sync you can stop it by pressing MENU and selecting Cancel sync.

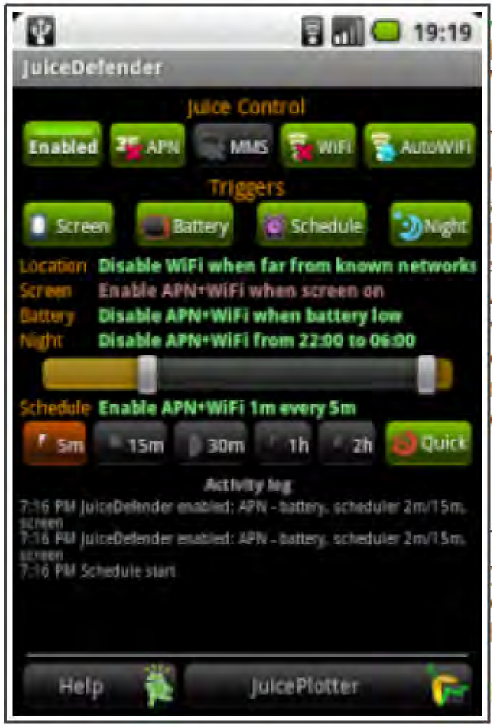
Sync problems If you see this icon ⟳! to the left of the sync check box, then there was a temporary problem with the data synchronization. Check your data connection and try again later.

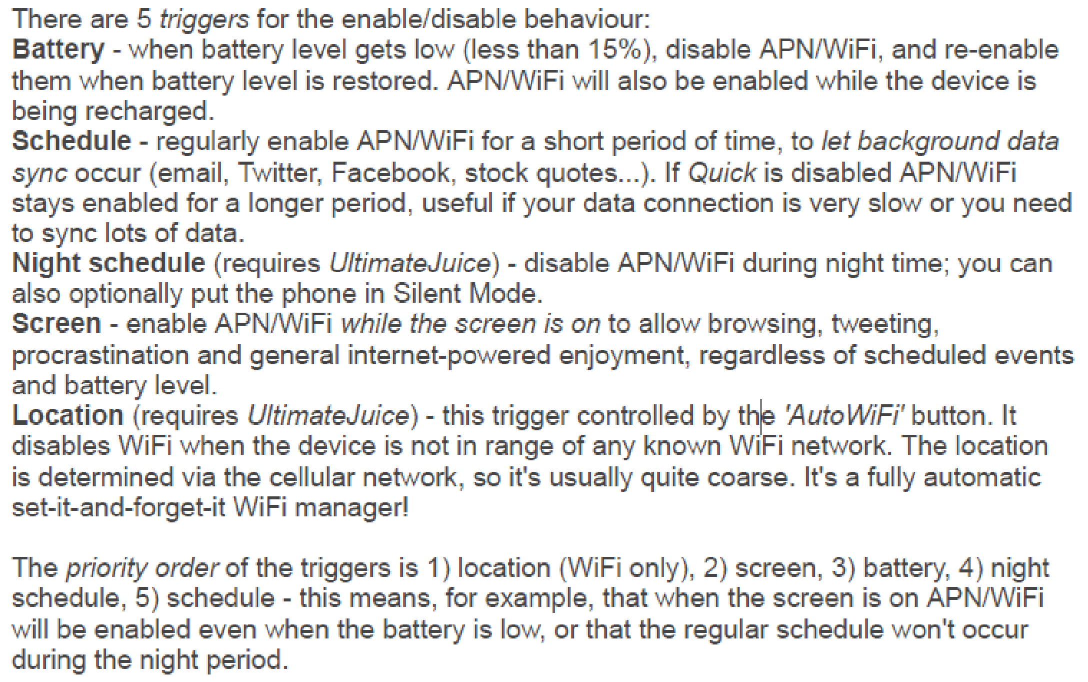
'701 Claims	Android Device with One or More Apps
	 **Sync by Gmail label** You can select to sync only Gmail messages with certain labels from the Gmail application settings screen. Read more in "Select labels to synchronize" on page 46. SAMSUNG_PRIORART0005245  Open in background Select if you want links to new pages to open in a new window in the background. SAMSUNG_PRIORART0005598

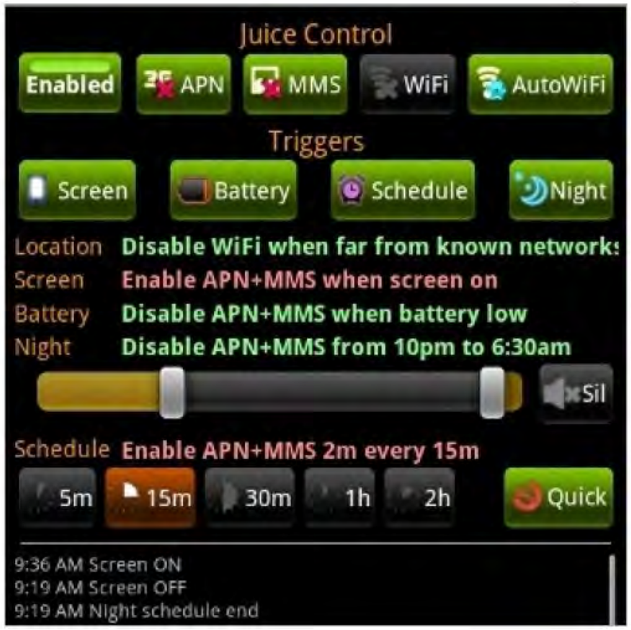
'701 Claims	Android Device with One or More Apps
	 8:08 PM Data synchronization General sync settings Background data ☑ Enable background data usage Auto-sync ☑ Automatically sync data Application sync settings Gmail ☑ 9/8/2011 3:30 AM Calendar ☑ 9/8/2011 3:30 AM Contacts ☑ 9/8/2011 3:29 AM **Samsung GT-I7500 Galaxy** SAMSUNG_PRIORART0005494 at PDF pgs. 38-39, 72:

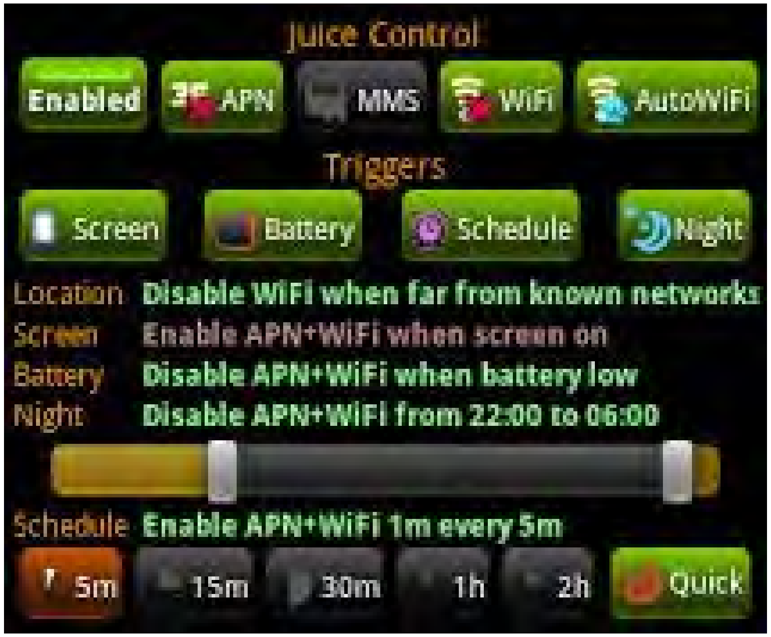
'701 Claims	Android Device with One or More Apps
	 **Synchronise data** You can synchronise data (Google messages, calendar, and contacts) with the Google web server and back up or restore your data. Once the synchronisation is completed, your device stays connected to the web. If any change is made on the web, the updated information will appear on your device and automatically will begin synchronisation, and vice versa. To synchronise with the Google web server, you must have a Google account. If you do not have a Google account, sign up for a Google account by selecting **Create** on the first setup screen. **Activate automatic sync** If you add or edit data in any of the applications (Google Mail, Calendar, and Contacts) on the web, the updated information will appear on your device, and vice versa. 1. From the Home screen, open the application list and select **Settings** → **Data synchronization**. 2. Select **Auto-sync**. 3. Select the applications you want to synchronise. To exclude applications from automatic synchronisation, clear the check box next to the application you want. **Synchronise data manually** 1. From the Home screen, open the application list and select **Settings** → **Data synchronization**. 2. Press [⏏] → **Sync now**. Your device will start synchronising the data you set for synchronisation.

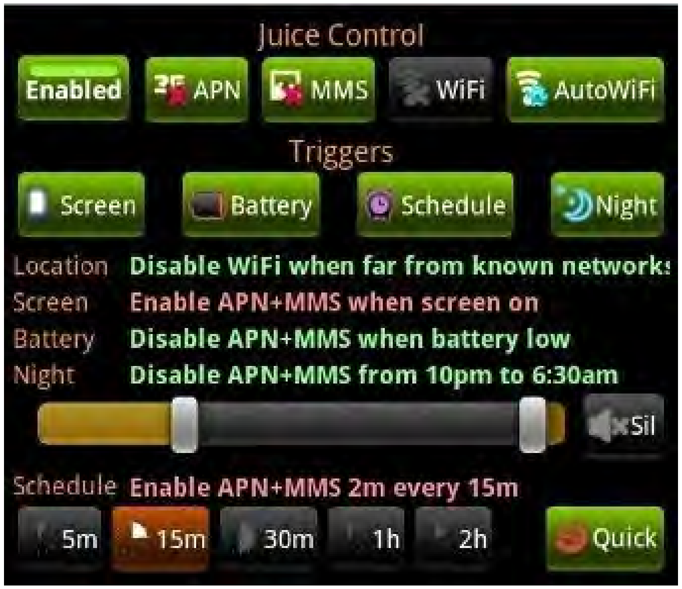
'701 Claims	Android Device with One or More Apps
	 Wi-Fi settings • **Wi-Fi**: Turn the WLAN feature on or off. • **Network notification**: Set the device to notify you when an open network is available. • **Add Wi-Fi network**: Add WLAN networks. Bluetooth Turn the Bluetooth wireless feature on or off. Bluetooth settings • **Bluetooth**: Turn the Bluetooth wireless feature on or off. • **Device name**: Set a Bluetooth name for your device. • **Discoverable**: Set your device to be visible to other Bluetooth devices. • **Scan for devices**: Search for available Bluetooth devices. Mobile networks • **Data roaming**: Set the device to connect another network when you are roaming or your home network is not available. • **Use only 2G networks**: Set the device to connect only to a 2G network. • **Network operators**: Search for available networks and select a network for roaming. • **Access Point Names**: Set up access point names (APNs). Airplane mode Disable all wireless functions on your device. **<u>JuiceDefender App</u>** SAMSUNG_PRIORART0000379 (Latedroid):

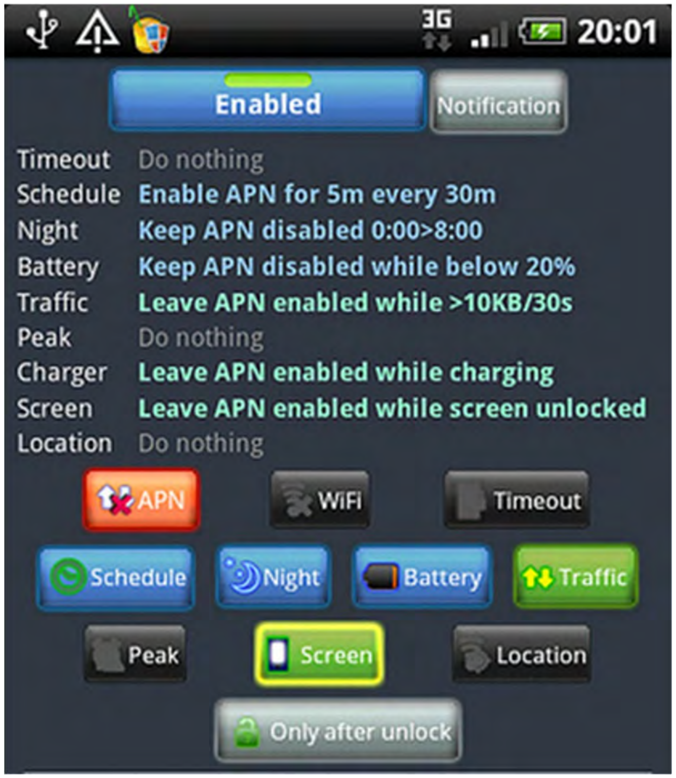
'701 Claims	Android Device with One or More Apps
	JuiceDefender  19:19 JuiceDefender Juice Control Enabled APN MMS WiFi AutoWiFi Triggers Screen Battery Schedule Night Location Disable WiFi when far from known networks Screen Enable APN+WiFi when screen on Battery Disable APN+WiFi when battery low Night Disable APN+WiFi from 22:00 to 06:00 Schedule Enable APN+WiFi 1m every 5m 5m 15m 30m 1h 2h Quick Activity log 7:16 PM JuiceDefender enabled: APN - battery, scheduler 2m/15m, screen 7:16 PM JuiceDefender enabled: APN - battery, scheduler 2m/15m, screen 7:16 PM Schedule start Help JuicePlotter JuiceDefender saves battery power (*lots of it!*) by controlling the device data connection and/or WiFi. You can schedule regular APN/WiFi activation to let background data sync occur and have APN/WiFi enabled while the screen is on. It also helps in minimizing distractions ;) The *Easy Mode* is a no-fuss one-click way to let your battery last longer - much longer. Just enable JuiceDefender by clicking on the big button and you're ready to go! If you want more fine-grained control, try *Advanced Mode*, where you can configure all JuiceDefender features to your liking.

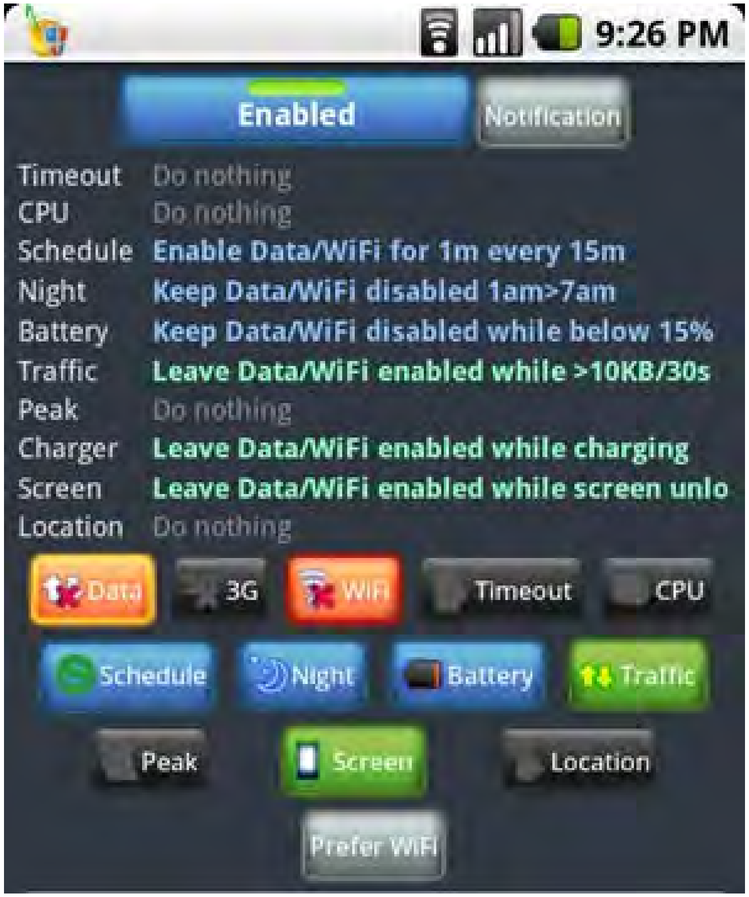
'701 Claims	Android Device with One or More Apps
	 There are 5 *triggers* for the enable/disable behaviour: **Battery** - when battery level gets low (less than 15%), disable APN/WiFi, and re-enable them when battery level is restored. APN/WiFi will also be enabled while the device is being recharged. **Schedule** - regularly enable APN/WiFi for a short period of time, to *let background data sync* occur (email, Twitter, Facebook, stock quotes...). If *Quick* is disabled APN/WiFi stays enabled for a longer period, useful if your data connection is very slow or you need to sync lots of data. **Night schedule** (requires *UltimateJuice*) - disable APN/WiFi during night time; you can also optionally put the phone in Silent Mode. **Screen** - enable APN/WiFi *while the screen is on* to allow browsing, tweeting, procrastination and general internet-powered enjoyment, regardless of scheduled events and battery level. **Location** (requires *UltimateJuice*) - this trigger controlled by the *'AutoWiFi'* button. It disables WiFi when the device is not in range of any known WiFi network. The location is determined via the cellular network, so it's usually quite coarse. It's a fully automatic set-it-and-forget-it WiFi manager! The *priority order* of the triggers is 1) location (WiFi only), 2) screen, 3) battery, 4) night schedule, 5) schedule - this means, for example, that when the screen is on APN/WiFi will be enabled even when the battery is low, or that the regular schedule won't occur during the night period. SAMSUNG_PRIORART0000361 (Purdy):

'701 Claims	Android Device with One or More Apps
	 Juice Control Enabled APN MMS WiFi AutoWiFi Triggers Screen Battery Schedule Night Sil 5m 15m 30m 1h 2h Quick Location Disable WiFi when far from known network: Screen Enable APN+MMS when screen on Battery Disable APN+MMS when battery low Night Disable APN+MMS from 10pm to 6:30am Schedule Enable APN+MMS 2m every 15m 9:36 AM Screen ON 9:19 AM Screen OFF 9:19 AM Night schedule end Android: Most phones don't make it easy to switch cellular data connection on and off, even if doing so really helps save your battery. JuiceDefender toggles wireless data and Wi-Fi on and off every so often to preserve power. The whole point of a smartphone with Google apps baked in is constant connectivity, of course, and you don't want to shut off access to your email, Google Voice messages, and other online services. But when you're walking around, at your office desk, and generally not actively using your phone, you probably don't need your phone to check in every minute with the mothership. JuiceDefender lets you set a time interval—5 minutes, 15, 30, an hour, two hours—at which its background process will re-enable your carrier APN, see if there are new messages or data coming in, and then shut off again. You can also set similar Wi-Fi connectivity rules, or only have web data enabled when you've got your screen on. Besides the battery savings, those who like to parse out their email checks and avoid minute-by-minute distractions see some benefit here, too. SAMSUNG_PRIORART0000379 (Latedroid):

'701 Claims	Android Device with One or More Apps
	 Juice Control Enabled APN MMS WiFi AutoWiFi Triggers Screen Battery Schedule Night 5m 15m 30m 1h 2h Quick Location Disable WiFi when far from known networks Screen Enable APN+WiFi when screen on Battery Disable APN+WiFi when battery low Night Disable APN+WiFi from 22:00 to 06:00 Schedule Enable APN+WiFi 1m every 5m SAMSUNG_PRIORART0000361 (Purdy):

’701 Claims	Android Device with One or More Apps
	 Juice Control Enabled APN MMS WiFi AutoWiFi Triggers Screen Battery Schedule Night Sil 5m 15m 30m 1h 2h Quick Location Disable WiFi when far from known network: Screen Enable APN+MMS when screen on Battery Disable APN+MMS when battery low Night Disable APN+MMS from 10pm to 6:30am Schedule Enable APN+MMS 2m every 15m SAMSUNG_PRIORART0000351 (Configuration-Translated):

'701 Claims	Android Device with One or More Apps
	 3G 20:01 Enabled Notification APN WiFi Timeout Schedule Night Battery Traffic Peak Screen Location Only after unlock Timeout Do nothing Schedule Enable APN for 5m every 30m Night Keep APN disabled 0:00>8:00 Battery Keep APN disabled while below 20% Traffic Leave APN enabled while >10KB/30s Peak Do nothing Charger Leave APN enabled while charging Screen Leave APN enabled while screen unlocked Location Do nothing HW_PRIOR_ART00002319 (Ruddock):

’701 Claims	Android Device with One or More Apps
	 9:26 PM Enabled Notification Data 3G WiFi Timeout CPU Schedule Night Battery Traffic Peak Screen Location Prefer WiFi Timeout Do nothing CPU Do nothing Schedule Enable Data/WiFi for 1m every 15m Night Keep Data/WiFi disabled 1am>7am Battery Keep Data/WiFi disabled while below 15% Traffic Leave Data/WiFi enabled while >10KB/30s Peak Do nothing Charger Leave Data/WiFi enabled while charging Screen Leave Data/WiFi enabled while screen unlo Location Do nothing **<u>GreenPower App</u>** POUZERATE0000196 (GreenPower User Guide) (“Manage Wifi

'701 Claims	Android Device with One or More Apps
	If this setting is selected, then Green Power will regularly turn on and off the Wifi connection, based on the durations specified in the settings below. If this setting is not selected, then Green Power will leave the Wifi as it is, never turning it on or off. Please note that if you manually switches off the Wifi, then Green Power will unselect the "Manage Wifi" setting in order not to automatically switch on the Wifi again despite your manual action. Then, if you switch back on the Wifi or reselect "Manage Wifi" setting, Green Power will resume managing Wifi connection. Manage Mobile Network If this setting is selected, then Green Power will regularly turn on and off the Mobile Network connection, based on the durations specified in the settings below. If this setting is not selected, then Green Power will leave the Mobile Network as it is, never turning it on or off. Please note that in order for Green Power to turn on / off Mobile Network, this one has to be manually enabled by the user first in the phone settings (Wireless & networks → Mobile Network) or in Green Power settings (Global wireless settings → Mobile Network) . Green Power can't itself turn on Mobile Network as this is a limitation of the Android system for security and cost reasons."). **Android 1.0** **SAMSUNG_PRIORART0005487, Activity** * <p>Activities in the system are managed as an <em>activity stack</em>. * When a new activity is started, it is placed on the top of the stack * and becomes the running activity -- the previous activity always remains

'701 Claims	Android Device with One or More Apps
	* below it in the stack, and will not come to the foreground again until * the new activity exits.</p> * * <p>An activity has essentially four states:</p> * <ul> * <li> If an activity in the foreground of the screen (at the top of * the stack), * it is <em>active</em> or <em>running</em>. </li> * <li>If an activity has lost focus but is still visible (that is, a new non-full-sized * or transparent activity has focus on top of your activity), it * is <em>paused</em>. A paused activity is completely alive (it * maintains all state and member information and remains attached to * the window manager), but can be killed by the system in extreme * low memory situations. * <li>If an activity is completely obscured by another activity, * it is <em>stopped</em>. It still retains all state and member information, * however, it is no longer visible to the user so its window is hidden * and it will often be killed by the system when memory is needed * elsewhere.</li> * <li>If an activity is paused or stopped, the system can drop the activity * from memory by either asking it to finish, or simply killing its * process. When it is displayed again to the user, it must be * completely restarted and restored to its previous state.</li> * </ul> * * <p>The following diagram shows the important state paths of an Activity. * The square rectangles represent callback methods you can implement to * perform operations when the Activity moves between states. The colored * ovals are major states the Activity can be in.</p> *

'701 Claims	Android Device with One or More Apps
	* <p><img src="../../../images/activity_lifecycle.png" <ul> * <li>The <b>entire lifetime</b> of an activity happens between the first call * to {@link android.app.Activity#onCreate} through to a single final call * to {@link android.app.Activity#onDestroy}. An activity will do all setup * of "global" state in onCreate(), and release all remaining resources in * onDestroy(). For example, if it has a thread running in the background * to download data from the network, it may create that thread in onCreate() * and then stop the thread in onDestroy(). * * <li>The <b>visible lifetime</b> of an activity happens between a call to * {@link android.app.Activity#onStart} until a corresponding call to * {@link android.app.Activity#onStop}. During this time the user can see the * activity on-screen, though it may not be in the foreground and interacting * with the user. Between these two methods you can maintain resources that * are needed to show the activity to the user. For example, you can register * a {@link android.content.BroadcastReceiver} in onStart() to monitor for changes * that impact your UI, and unregister it in onStop() when the user an no * longer see what you are displaying. The onStart() and onStop() methods * can be called multiple times, as the activity becomes visible and hidden * to the user. * * <li>The <b>foreground lifetime</b> of an activity happens between a call to * {@link android.app.Activity#onResume} until a corresponding call to * {@link android.app.Activity#onPause}. During this time the activity is * in front of all other activities and interacting with the user. An activity * can frequently go between the resumed and paused states -- for example when * the device goes to sleep, when an activity result is delivered, when a new * intent is delivered -- so the code in these methods should be fairly

'701 Claims	Android Device with One or More Apps
	* lightweight. * </ul> * * <p>The entire lifecycle of an activity is defined by the following * Activity methods. All of these are hooks that you can override * to do appropriate work when the activity changes state. All * activities will implement {@link android.app.Activity#onCreate} * to do their initial setup; many will also implement * {@link android.app.Activity#onPause} to commit changes to data and * otherwise prepare to stop interacting with the user. You should always * call up to your superclass when implementing these methods.</p> * <tr><th colspan="2" align="left" border="0">{@link android.app.Activity#onStop onStop()}</th> * <td>Called when the activity is no longer visible to the user, because * another activity has been resumed and is covering this one. This * may happen either because a new activity is being started, an existing * one is being brought in front of this one, or this one is being * destroyed. * <p>Followed by either <code>onRestart()</code> if * this activity is coming back to interact with the user, or * <code>onDestroy()</code> if this activity is going away.</td> * <td align="center"><font color="#800000"><strong>Yes</strong></font></td> * <td align="center"><code>onRestart()</code> or<br><br>* <code>onDestroy()</code></td> * </tr> * <p>The Android system attempts to keep application process around for as * long as possible, but eventually will need to remove old processes when

'701 Claims	Android Device with One or More Apps
	* memory runs low. As described in <a href="#ActivityLifecycle">Activity * Lifecycle</a>, the decision about which process to remove is intimately * tied to the state of the user's interaction with it. In general, there * are four states a process can be in based on the activities running in it, * listed here in order of importance. The system will kill less important * processes (the last ones) before it resorts to killing more important * processes (the first ones). * * <ol> * <li> <p>The <b>foreground activity</b> (the activity at the top of the screen * that the user is currently interacting with) is considered the most important. * Its process will only be killed as a last resort, if it uses more memory * than is available on the device. Generally at this point the device has * reached a memory paging state, so this is required in order to keep the user * interface responsive. * <li> <p>A <b>visible activity</b> (an activity that is visible to the user * but not in the foreground, such as one sitting behind a foreground dialog) * is considered extremely important and will not be killed unless that is * required to keep the foreground activity running. * <li> <p>A <b>background activity</b> (an activity that is not visible to * the user and has been paused) is no longer critical, so the system may * safely kill its process to reclaim memory for other foreground or * visible processes. If its process needs to be killed, when the user navigates * back to the activity (making it visible on the screen again), its * {@link #onCreate} method will be called with the savedInstanceState it had previously * supplied in {@link #onSaveInstanceState} so that it can restart itself in the same * state as the user last left it. * <li> <p>An <b>empty process</b> is one hosting no activities or other

'701 Claims	Android Device with One or More Apps
	* application components (such as {@link Service} or * {@link android.content.BroadcastReceiver} classes). These are killed very * quickly by the system as memory becomes low. For this reason, any * background operation you do outside of an activity must be executed in the * context of an activity BroadcastReceiver or Service to ensure that the system * knows it needs to keep your process around. * </ol> * /** * Called as part of the activity lifecycle when an activity is going into * the background, but has not (yet) been killed. The counterpart to * {@link #onResume}. * * <p>When activity B is launched in front of activity A, this callback will * be invoked on A. B will not be created until A's {@link #onPause} returns, * so be sure to not do anything lengthy here. * * <p>This callback is mostly used for saving any persistent state the * activity is editing, to present a "edit in place" model to the user and * making sure nothing is lost if there are not enough resources to start * the new activity without first killing this one. This is also a good * place to do things like stop animations and other things that consume a * noticeable mount of CPU in order to make the switch to the next activity * as fast as possible, or to close resources that are exclusive access * such as the camera. * * <p>In situations where the system needs more memory it may kill paused * processes to reclaim resources. Because of this, you should be sure

'701 Claims	Android Device with One or More Apps
	* that all of your state is saved by the time you return from * this function. In general {@link #onSaveInstanceState} is used to save * per-instance state in the activity and this method is used to store * global persistent data (in content providers, files, etc.) * * <p>After receiving this call you will usually receive a following call * to {@link #onStop} (after the next activity has been resumed and * displayed), however in some cases there will be a direct call back to * {@link #onResume} without going through the stopped state. * * <p><em>Derived classes must call through to the super class's * implementation of this method. If they do not, an exception will be * thrown.</em></p> * * @see #onResume * @see #onSaveInstanceState * @see #onStop */ **SAMSUNG_PRIORART0005487, ActivityManager** /** * Return a list of the tasks that are currently running, with * the most recent being first and older ones after in order. Note that * "running" does not mean any of the task's code is currently loaded or * activity -- the task may have been frozen by the system, so that it * can be restarted in its previous state when next brought to the * foreground. * * @param maxNum The maximum number of entries to return in the list. The

<table>
<tr><th>'701 Claims</th><th>Android Device with One or More Apps</th></tr>
<tr><td></td><td><pre>
     * actual number returned may be smaller, depending on how many tasks the
     * user has started.
     *
     * @return Returns a list of RunningTaskInfo records describing each of
     * the running tasks.
     *
     * @throws SecurityException Throws SecurityException if the caller does
     * not hold the {@link android.Manifest.permission#GET_TASKS} permission.
     */
    public List<RunningTaskInfo> getRunningTasks(int maxNum)
            throws SecurityException {
        try {
            return (List<RunningTaskInfo>)ActivityManagerNative.getDefault()
                    .getTasks(maxNum, 0, null);
        } catch (RemoteException e) {
            // System dead, we will be dead too soon!
            return null;
        }
    }

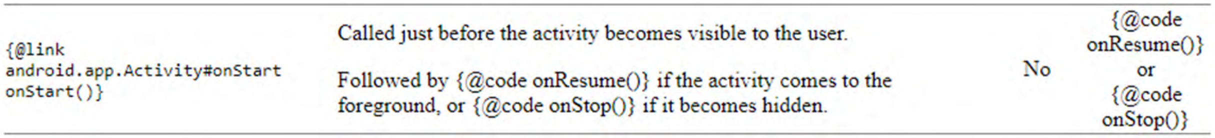
        /**
         * Set to true if the service has asked to run as a foreground process.
         */
        public boolean foreground;
</pre>

Android 1.6</td></tr>
</table>

'701 Claims	Android Device with One or More Apps
	SAMSUNG_PRIORART0005350, Fundamentals.jd **Activities** An *activity* presents a visual user interface for one focused endeavor the user can undertake. For example, an activity might present a list of menu items users can choose from or it might display photographs along with their captions. A text messaging application might have one activity that shows a list of contacts to send messages to, a second activity to write the message to the chosen contact, and other activities to review old messages or change settings. Though they work together to form a cohesive user interface, each activity is independent of the others. Each one is implemented as a subclass of the {@link android.app.Activity} base class. An application might consist of just one activity or, like the text messaging application just mentioned, it may contain several. What the activities are, and how many there are depends, of course, on the application and its design. Typically, one of the activities is marked as the first one that should be presented to the user when the application is launched. Moving from one activity to another is accomplished by having the current activity start the next one. Each activity is given a default window to draw in. Typically, the window fills the screen, but it might be smaller than the screen and float on top of other windows. An activity can also make use of additional windows — for example, a pop-up dialog that calls for a user response in the midst of the activity, or a window that presents users with vital information when they select a particular item on-screen. The visual content of the window is provided by a hierarchy of views — objects derived from the base {@link android.view.View} class. Each view controls a particular rectangular space within the window. Parent views contain and organize the layout of their children. Leaf views (those at the bottom of the hierarchy) draw in the rectangles they control and respond to user actions directed at that space. Thus, views are where the activity's interaction with the user takes place. For example, a view might display a small image and initiate an action when the user taps that image. Android has a number of ready-made views that you can use — including buttons, text fields, scroll bars, menu items, check boxes, and more.

’701 Claims	Android Device with One or More Apps
	A view hierarchy is placed within an activity's window by the {@link android.app.Activity#setContentView Activity.setContentView()} method. The *content view* is the View object at the root of the hierarchy. (See the separate User Interface document for more information on views and the hierarchy.) **Services** A *service* doesn't have a visual user interface, but rather runs in the background for an indefinite period of time. For example, a service might play background music as the user attends to other matters, or it might fetch data over the network or calculate something and provide the result to activities that need it. Each service extends the {@link android.app.Service} base class. A prime example is a media player playing songs from a play list. The player application would probably have one or more activities that allow the user to choose songs and start playing them. However, the music playback itself would not be handled by an activity because users will expect the music to keep playing even after they leave the player and begin something different. To keep the music going, the media player activity could start a service to run in the background. The system would then keep the music playback service running even after the activity that started it leaves the screen. It's possible to connect to (bind to) an ongoing service (and start the service if it's not already running). While connected, you can communicate with the service through an interface that the service exposes. For the music service, this interface might allow users to pause, rewind, stop, and restart the playback. Like activities and the other components, services run in the main thread of the application process. So that they won't block other components or the user interface, they often spawn another thread for time-consuming tasks (like music playback). See Processes and Threads, later. All the activities in a task move together as a unit. The entire task (the entire activity stack) can be brought to the foreground or sent to the background. Suppose, for instance, that the current task has four activities in its stack — three under the current activity. The user presses the HOME key, goes to the application launcher, and selects a new application (actually, a new *task*). The current task goes into the background and the root activity for the new task is displayed. Then, after a short period, the user goes

'701 Claims	Android Device with One or More Apps
	back to the home screen and again selects the previous application (the previous task). That task, with all four activities in the stack, comes forward. When the user presses the BACK key, the screen does not display the activity the user just left (the root activity of the previous task). Rather, the activity on the top of the stack is removed and the previous activity in the same task is displayed. As noted above, there's never more than one instance of a "{@code singleTask}" or "{@code singleInstance}" activity, so that instance is expected to handle all new intents. A "{@code singleInstance}" activity is always at the top of the stack (since it is the only activity in the task), so it is always in position to handle the intent. However, a "{@code singleTask}" activity may or may not have other activities above it in the stack. If it does, it is not in position to handle the intent, and the intent is dropped. (Even though the intent is dropped, its arrival would have caused the task to come to the foreground, where it would remain.) An activity has essentially three states: • It is *active* or *running* when it is in the foreground of the screen (at the top of the activity stack for the current task). This is the activity that is the focus for the user's actions. • It is *paused* if it has lost focus but is still visible to the user. That is, another activity lies on top of it and that activity either is transparent or doesn't cover the full screen, so some of the paused activity can show through. A paused activity is completely alive (it maintains all state and member information and remains attached to the window manager), but can be killed by the system in extreme low memory situations. • It is *stopped* if it is completely obscured by another activity. It still retains all state and member information. However, it is no longer visible to the user so its window is hidden and it will often be killed by the system when memory is needed elsewhere. Taken together, these seven methods define the entire lifecycle of an activity. There are three nested loops that you can monitor by implementing them: • The **entire lifetime** of an activity happens between the first call to {@link android.app.Activity#onCreate onCreate()} through to a single final call to {@link android.app.Activity#onDestroy}. An activity does all its initial setup of "global" state in {@code

'701 Claims	Android Device with One or More Apps
	onCreate()}, and releases all remaining resources in {@code onDestroy()}. For example, if it has a thread running in the background to download data from the network, it may create that thread in {@code onCreate()} and then stop the thread in {@code onDestroy()}. • The **visible lifetime** of an activity happens between a call to {@link android.app.Activity#onStart onStart()} until a corresponding call to {@link android.app.Activity#onStop onStop()}. During this time, the user can see the activity on-screen, though it may not be in the foreground and interacting with the user. Between these two methods, you can maintain resources that are needed to show the activity to the user. For example, you can register a {@link android.content.BroadcastReceiver} in {@code onStart()} to monitor for changes that impact your UI, and unregister it in {@code onStop()} when the user can no longer see what you are displaying. The {@code onStart()} and {@code onStop()} methods can be called multiple times, as the activity alternates between being visible and hidden to the user. • The **foreground lifetime** of an activity happens between a call to {@link android.app.Activity#onResume onResume()} until a corresponding call to {@link android.app.Activity#onPause onPause()}. During this time, the activity is in front of all other activities on screen and is interacting with the user. An activity can frequently transition between the resumed and paused states — for example, {@code onPause()} is called when the device goes to sleep or when a new activity is started, {@code onResume()} is called when an activity result or a new intent is delivered. Therefore, the code in these two methods should be fairly lightweight.  {@link android.app.Activity#onStart onStart()} | Called just before the activity becomes visible to the user. Followed by {@code onResume()} if the activity comes to the foreground, or {@code onStop()} if it becomes hidden. | No | {@code onResume()} or {@code onStop()} Processes and lifecycles The Android system tries to maintain an application process for as long as possible, but eventually it will need to remove old processes when memory runs low. To determine which processes to keep and which to kill, Android places each process into an "importance hierarchy" based on the components running in

'701 Claims	Android Device with One or More Apps
	it and the state of those components. Processes with the lowest importance are eliminated first, then those with the next lowest, and so on. There are five levels in the hierarchy. The following list presents them in order of importance: 1. A **foreground process** is one that is required for what the user is currently doing. A process is considered to be in the foreground if any of the following conditions hold: o It is running an activity that the user is interacting with (the Activity object's {@link android.app.Activity#onResume onResume()} method has been called). o It hosts a service that's bound to the activity that the user is interacting with. o It has a {@link android.app.Service} object that's executing one of its lifecycle callbacks ({@link android.app.Service#onCreate onCreate()}, {@link android.app.Service#onStart onStart()}, or {@link android.app.Service#onDestroy onDestroy()}). o It has a {@link android.content.BroadcastReceiver} object that's executing its {@link android.content.BroadcastReceiver#onReceive onReceive()} method. Only a few foreground processes will exist at any given time. They are killed only as a last resort — if memory is so low that they cannot all continue to run. Generally, at that point, the device has reached a memory paging state, so killing some foreground processes is required to keep the user interface responsive. 2. A **visible process** is one that doesn't have any foreground components, but still can affect what the user sees on screen. A process is considered to be visible if either of the following conditions holds: o It hosts an activity that is not in the foreground, but is still visible to the user (its {@link android.app.Activity#onPause onPause()} method has been called). This may occur, for example, if the foreground activity is a dialog that allows the previous activity to be seen behind it. o It hosts a service that's bound to a visible activity. A visible process is considered extremely important and will not be killed unless doing so is required to keep all foreground processes running. 3. A **service process** is one that is running a service that has been started with the {@link android.content.Context#startService startService()} method and that does not fall into either of the two higher categories. Although service processes are not directly tied to anything the user

’701 Claims	Android Device with One or More Apps
	sees, they are generally doing things that the user cares about (such as playing an mp3 in the background or downloading data on the network), so the system keeps them running unless there's not enough memory to retain them along with all foreground and visible processes. 4. A **background process** is one holding an activity that's not currently visible to the user (the Activity object's {@link android.app.Activity#onStop onStop()} method has been called). These processes have no direct impact on the user experience, and can be killed at any time to reclaim memory for a foreground, visible, or service process. Usually there are many background processes running, so they are kept in an LRU (least recently used) list to ensure that the process with the activity that was most recently seen by the user is the last to be killed. If an activity implements its lifecycle methods correctly, and captures its current state, killing its process will not have a deleterious effect on the user experience. 5. An **empty process** is one that doesn't hold any active application components. The only reason to keep such a process around is as a cache to improve startup time the next time a component needs to run in it. The system often kills these processes in order to balance overall system resources between process caches and the underlying kernel caches. **SAMSUNG_PRIORART0005487, Activity** * <p>An activity has essentially four states:</p> * <ul> * <li> If an activity in the foreground of the screen (at the top of * the stack), * it is <em>active</em> or <em>running</em>. </li> * <li>If an activity has lost focus but is still visible (that is, a new non-full-sized * or transparent activity has focus on top of your activity), it * is <em>paused</em>. A paused activity is completely alive (it * maintains all state and member information and remains attached to * the window manager), but can be killed by the system in extreme * low memory situations. * <li>If an activity is completely obscured by another activity, * it is <em>stopped</em>. It still retains all state and member information,

’701 Claims	Android Device with One or More Apps
	* however, it is no longer visible to the user so its window is hidden * and it will often be killed by the system when memory is needed * elsewhere.</li> * <li>If an activity is paused or stopped, the system can drop the activity * from memory by either asking it to finish, or simply killing its * process. When it is displayed again to the user, it must be * completely restarted and restored to its previous state.</li> * </ul> <li>The <b>visible lifetime</b> of an activity happens between a call to * {@link android.app.Activity#onStart} until a corresponding call to * {@link android.app.Activity#onStop}. During this time the user can see the * activity on-screen, though it may not be in the foreground and interacting * with the user. Between these two methods you can maintain resources that * are needed to show the activity to the user. For example, you can register * a {@link android.content.BroadcastReceiver} in onStart() to monitor for changes * that impact your UI, and unregister it in onStop() when the user an no * longer see what you are displaying. The onStart() and onStop() methods * can be called multiple times, as the activity becomes visible and hidden * to the user. * * <li>The <b>foreground lifetime</b> of an activity happens between a call to * {@link android.app.Activity#onResume} until a corresponding call to * {@link android.app.Activity#onPause}. During this time the activity is * in front of all other activities and interacting with the user. An activity * can frequently go between the resumed and paused states -- for example when * the device goes to sleep, when an activity result is delivered, when a new * intent is delivered -- so the code in these methods should be fairly * lightweight. * </ul>

’701 Claims	Android Device with One or More Apps
	* <p>Here is an excerpt from a calendar activity that stores the user's * preferred view mode in its persistent settings:</p> * * <pre class="prettyprint"> * public class CalendarActivity extends Activity { * ... * * static final int DAY_VIEW_MODE = 0; * static final int WEEK_VIEW_MODE = 1; <ol> * <li> <p>The <b>foreground activity</b> (the activity at the top of the screen * that the user is currently interacting with) is considered the most important. * Its process will only be killed as a last resort, if it uses more memory * than is available on the device. Generally at this point the device has * reached a memory paging state, so this is required in order to keep the user * interface responsive. * <li> <p>A <b>visible activity</b> (an activity that is visible to the user * but not in the foreground, such as one sitting behind a foreground dialog) * is considered extremely important and will not be killed unless that is * required to keep the foreground activity running. * <li> <p>A <b>background activity</b> (an activity that is not visible to * the user and has been paused) is no longer critical, so the system may * safely kill its process to reclaim memory for other foreground or * visible processes. If its process needs to be killed, when the user navigates * back to the activity (making it visible on the screen again), its * {@link #onCreate} method will be called with the savedInstanceState it had previously * supplied in {@link #onSaveInstanceState} so that it can restart itself in the same * state as the user last left it.

'701 Claims	Android Device with One or More Apps
	* <li> <p>An <b>empty process</b> is one hosting no activities or other * application components (such as {@link Service} or * {@link android.content.BroadcastReceiver} classes). These are killed very * quickly by the system as memory becomes low. For this reason, any * background operation you do outside of an activity must be executed in the * context of an activity BroadcastReceiver or Service to ensure that the system * knows it needs to keep your process around. * </ol> * * <p>Sometimes an Activity may need to do a long-running operation that exists * independently of the activity lifecycle itself. An example may be a camera * application that allows you to upload a picture to a web site. The upload * may take a long time, and the application should allow the user to leave * the application will it is executing. To accomplish this, your Activity * should start a {@link Service} in which the upload takes place. This allows * the system to properly prioritize your process (considering it to be more * important than other non-visible applications) for the duration of the * upload, independent of whether the original activity is paused, stopped, * or finished. */ **SAMSUNG_PRIORART0005350, ActivityManager** /** * Return a list of the tasks that the user has recently launched, with * the most recent being first and older ones after in order. * * @param maxNum The maximum number of entries to return in the list. The * actual number returned may be smaller, depending on how many tasks the * user has started and the maximum number the system can remember.

<table>
<tr><th>'701 Claims</th><th>Android Device with One or More Apps</th></tr>
<tr><td></td><td><pre>
     *
     * @return Returns a list of RecentTaskInfo records describing each of
     * the recent tasks.
     *
     * @throws SecurityException Throws SecurityException if the caller does
     * not hold the {@link android.Manifest.permission#GET_TASKS} permission.
     */
    public List<RecentTaskInfo> getRecentTasks(int maxNum, int flags)
            throws SecurityException {
        try {
            return ActivityManagerNative.getDefault().getRecentTasks(maxNum,
                    flags);
        } catch (RemoteException e) {
            // System dead, we will be dead too soon!
            return null;
        }
    }

/**
     * Return a list of the tasks that are currently running, with
     * the most recent being first and older ones after in order. Note that
     * "running" does not mean any of the task's code is currently loaded or
     * activity -- the task may have been frozen by the system, so that it
     * can be restarted in its previous state when next brought to the
     * foreground.
     *
     * @param maxNum The maximum number of entries to return in the list. The
     * actual number returned may be smaller, depending on how many tasks the
     * user has started.
     *
     * @return Returns a list of RunningTaskInfo records describing each of
</pre></td></tr>
</table>

'701 Claims	Android Device with One or More Apps

	* the running tasks.
	*
	* @throws SecurityException Throws SecurityException if the caller does
	* not hold the {@link android.Manifest.permission#GET_TASKS} permission.
	*/
	public List<RunningTaskInfo> getRunningTasks(int maxNum)
	throws SecurityException {
	try {
	return (List<RunningTaskInfo>)ActivityManagerNative.getDefault()
	.getTasks(maxNum, 0, null);
	} catch (RemoteException e) {
	// System dead, we will be dead too soon!
	return null;
	}
	}
	
	/**
	* Set to true if the service has asked to run as a foreground process.
	*/
	public boolean foreground;
	
	/**
	* Return a list of the services that are currently running.
	*
	* @param maxNum The maximum number of entries to return in the list. The
	* actual number returned may be smaller, depending on how many services
	* are running.
	*
	* @return Returns a list of RunningServiceInfo records describing each of
	* the running tasks.

<table>
<tr><th>'701 Claims</th><th>Android Device with One or More Apps</th></tr>
<tr><td></td><td><pre>
     */
    public List<RunningServiceInfo> getRunningServices(int maxNum)
            throws SecurityException {
        try {
            return (List<RunningServiceInfo>)ActivityManagerNative.getDefault()
                    .getServices(maxNum, 0);
        } catch (RemoteException e) {
            // System dead, we will be dead too soon!
            return null;
        }
    }

    /**
     * Constant for {@link #importance}: this process is running the
     * foreground UI.
     */
    public static final int IMPORTANCE_FOREGROUND = 100;

    /**
     * Constant for {@link #importance}: this process is running something
     * that is considered to be actively visible to the user.
     */
    public static final int IMPORTANCE_VISIBLE = 200;

    /**
     * Constant for {@link #importance}: this process is contains services
     * that should remain running.
     */
    public static final int IMPORTANCE_SERVICE = 300;

    /**
</pre></td></tr>
</table>

’701 Claims	Android Device with One or More Apps

	* Constant for {@link #importance}: this process process contains
	    * background code that is expendable.
	    */
	   public static final int IMPORTANCE_BACKGROUND = 400;
	
	/**
	    * The relative importance level that the system places on this
	    * process. May be one of {@link #IMPORTANCE_FOREGROUND},
	    * {@link #IMPORTANCE_VISIBLE}, {@link #IMPORTANCE_SERVICE},
	    * {@link #IMPORTANCE_BACKGROUND}, or {@link #IMPORTANCE_EMPTY}. These
	    * constants are numbered so that "more important" values are always
	    * smaller than "less important" values.
	    */
	   public int importance;
	
	/**
	    * An additional ordering within a particular {@link #importance}
	    * category, providing finer-grained information about the relative
	    * utility of processes within a category. This number means nothing
	    * except that a smaller values are more recently used (and thus
	    * more important). Currently an LRU value is only maintained for
	    * the {@link #IMPORTANCE_BACKGROUND} category, though others may
	    * be maintained in the future.
	    */
	   public int lru;
	
	   public RunningAppProcessInfo() {
	       importance = IMPORTANCE_FOREGROUND;
	   }
	
	/**

'701 Claims	Android Device with One or More Apps

	* Returns a list of application processes that are running on the device.
	*
	* @return Returns a list of RunningAppProcessInfo records, or null if there are no
	* running processes (it will not return an empty list). This list ordering is not
	* specified.
	*/
	public List<RunningAppProcessInfo> getRunningAppProcesses() {
	try {
	return ActivityManagerNative.getDefault().getRunningAppProcesses();
	} catch (RemoteException e) {
	return null;
	}
	}
	
	SAMSUNG_PRIORART0005350, ConnectivityManager
	
	/**
	* Class that answers queries about the state of network connectivity. It also
	* notifies applications when network connectivity changes. Get an instance
	* of this class by calling
	* {@link android.content.Context#getSystemService(String)
	Context.getSystemService(Context.CONNECTIVITY_SERVICE)}.
	* <p>
	* The primary responsibilities of this class are to:
	* <ol>
	* <li>Monitor network connections (Wi-Fi, GPRS, UMTS, etc.)</li>
	* <li>Send broadcast intents when network connectivity changes</li>
	* <li>Attempt to "fail over" to another network when connectivity to a network
	* is lost</li>
	* <li>Provide an API that allows applications to query the coarse-grained or fine-grained
	* state of the available networks</li>

'701 Claims	Android Device with One or More Apps

```
 * </ol>
 */

@SdkConstant(SdkConstantType.BROADCAST_INTENT_ACTION)
    public static final String ACTION_BACKGROUND_DATA_SETTING_CHANGED =
            "android.net.conn.BACKGROUND_DATA_SETTING_CHANGED";

    public static final int TYPE_MOBILE = 0;
    public static final int TYPE_WIFI   = 1;

    public static final int DEFAULT_NETWORK_PREFERENCE = TYPE_WIFI;

static public boolean isNetworkTypeValid(int networkType) {
        return networkType == TYPE_WIFI || networkType == TYPE_MOBILE;
    }

    public void setNetworkPreference(int preference) {
        try {
            mService.setNetworkPreference(preference);
        } catch (RemoteException e) {
        }
    }

    /** {@hide} */
    public boolean setRadio(int networkType, boolean turnOn) {
        try {
            return mService.setRadio(networkType, turnOn);
        } catch (RemoteException e) {
            return false;
        }
    }
```

'701 Claims	Android Device with One or More Apps

```
/**
 * Returns the value of the setting for background data usage. If false,
 * applications should not use the network if the application is not in the
 * foreground. Developers should respect this setting, and check the value
 * of this before performing any background data operations.
 * <p>
 * All applications that have background services that use the network
 * should listen to {@link #ACTION_BACKGROUND_DATA_SETTING_CHANGED}.
 *
 * @return Whether background data usage is allowed.
 */
public boolean getBackgroundDataSetting() {
    try {
        return mService.getBackgroundDataSetting();
    } catch (RemoteException e) {
        // Err on the side of safety
        return false;
    }
}

/**
 * Sets the value of the setting for background data usage.
 *
 * @param allowBackgroundData Whether an application should use data while
 *            it is in the background.
 *
 * @attr ref android.Manifest.permission#CHANGE_BACKGROUND_DATA_SETTING
 * @see #getBackgroundDataSetting()
 * @hide
 */
```

’701 Claims	Android Device with One or More Apps

	public void setBackgroundDataSetting(boolean allowBackgroundData) {
	try {
	mService.setBackgroundDataSetting(allowBackgroundData);
	} catch (RemoteException e) {
	}
	}
	
	SAMSUNG_PRIORART0005350, ConnectivityService
	
	/*
	* Create the network state trackers for Wi-Fi and mobile
	* data. Maybe this could be done with a factory class,
	* but it's not clear that it's worth it, given that
	* the number of different network types is not going
	* to change very often.
	*/
	if (DBG) Log.v(TAG, "Starting Wifi Service.");
	mWifiStateTracker = new WifiStateTracker(context, handler);
	WifiService wifiService = new WifiService(context, mWifiStateTracker);
	ServiceManager.addService(Context.WIFI_SERVICE, wifiService);
	mNetTrackers[ConnectivityManager.TYPE_WIFI] = mWifiStateTracker;
	
	mMobileDataStateTracker = new MobileDataStateTracker(context, handler);
	mNetTrackers[ConnectivityManager.TYPE_MOBILE] = mMobileDataStateTracker;
	
	/**
	* Make the state of network connectivity conform to the preference settings.
	* In this method, we only tear down a non-preferred network. Establishing
	* a connection to the preferred network is taken care of when we handle
	* the disconnect event from the non-preferred network

'701 Claims	Android Device with One or More Apps
	(code below)

```
   * (see {@link #handleDisconnect(NetworkInfo)}).
   */
  private void enforcePreference() {
    if (mActiveNetwork == null)
      return;

    for (NetworkStateTracker t : mNetTrackers) {
      if (t == mActiveNetwork) {
        int netType = t.getNetworkInfo().getType();
        int otherNetType = ((netType == ConnectivityManager.TYPE_WIFI) ?
            ConnectivityManager.TYPE_MOBILE :
            ConnectivityManager.TYPE_WIFI);

        if (t.getNetworkInfo().getType() != mNetworkPreference) {
          NetworkStateTracker otherTracker = mNetTrackers[otherNetType];
          if (otherTracker.isAvailable()) {
            teardown(t);
          }
        }
      }
    }
  }

/**
   * @see ConnectivityManager#getBackgroundDataSetting()
   */
  public boolean getBackgroundDataSetting() {
    return Settings.Secure.getInt(mContext.getContentResolver(),
        Settings.Secure.BACKGROUND_DATA, 1) == 1;
  }
```

<table>
<tr><th>'701 Claims</th><th>Android Device with One or More Apps</th></tr>
<tr><td></td><td><pre>
  /**
   * @see ConnectivityManager#setBackgroundDataSetting(boolean)
   */
  public void setBackgroundDataSetting(boolean allowBackgroundDataUsage) {
      mContext.enforceCallingOrSelfPermission(
              android.Manifest.permission.CHANGE_BACKGROUND_DATA_SETTING,
              "ConnectivityService");

      if (getBackgroundDataSetting() == allowBackgroundDataUsage) return;

      Settings.Secure.putInt(mContext.getContentResolver(),
              Settings.Secure.BACKGROUND_DATA, allowBackgroundDataUsage ? 1 : 0);

      Intent broadcast = new Intent(
              ConnectivityManager.ACTION_BACKGROUND_DATA_SETTING_CHANGED);
      mContext.sendBroadcast(broadcast);
  }

       /**
        * See if the other network is available to fail over to.
        * If is not available, we enable it anyway, so that it
        * will be able to connect when it does become available,
        * but we report a total loss of connectivity rather than
        * report that we are attempting to fail over.
        */
      NetworkInfo switchTo = null;
      if (newNet.isAvailable()) {
          mActiveNetwork = newNet;
          switchTo = newNet.getNetworkInfo();
          switchTo.setFailover(true);
</pre></td></tr>
</table>

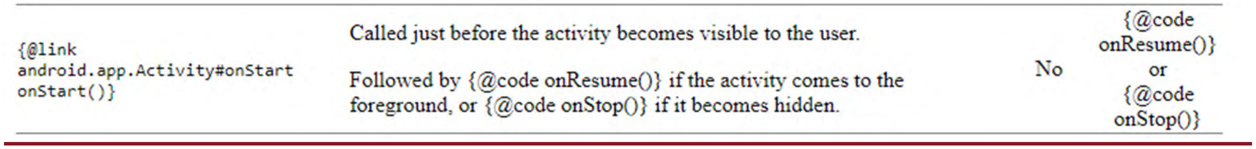
'701 Claims	Android Device with One or More Apps

	if (!switchTo.isConnectedOrConnecting()) {
	            newNet.reconnect();
	        }
	    } else {
	        newNet.reconnect();
	    }
	
	    if (info.getType() == ConnectivityManager.TYPE_MOBILE) {
	        otherNet = mWifiStateTracker;
	    } else /* info().getType() == TYPE_WIFI */ {
	        otherNet = mMobileDataStateTracker;
	    }
	
	    int incrValue = ConnectivityManager.TYPE_MOBILE - ConnectivityManager.TYPE_WIFI;
	    int stopValue = ConnectivityManager.TYPE_MOBILE + incrValue;
	
	Android 2.2
	
	SAMSUNG_PRIORART0005350, Fundamentals.jd
	
	Activities
	
	An *activity* presents a visual user interface for one focused endeavor the user can undertake. For example, an activity might present a list of menu items users can choose from or it might display photographs along with their captions. A text messaging application might have one activity that shows a list of contacts to send messages to, a second activity to write the message to the chosen contact, and

'701 Claims	Android Device with One or More Apps
	other activities to review old messages or change settings. Though they work together to form a cohesive user interface, each activity is independent of the others. Each one is implemented as a subclass of the {@link android.app.Activity} base class. An application might consist of just one activity or, like the text messaging application just mentioned, it may contain several. What the activities are, and how many there are depends, of course, on the application and its design. Typically, one of the activities is marked as the first one that should be presented to the user when the application is launched. Moving from one activity to another is accomplished by having the current activity start the next one. Each activity is given a default window to draw in. Typically, the window fills the screen, but it might be smaller than the screen and float on top of other windows. An activity can also make use of additional windows — for example, a pop-up dialog that calls for a user response in the midst of the activity, or a window that presents users with vital information when they select a particular item on-screen. The visual content of the window is provided by a hierarchy of views — objects derived from the base {@link android.view.View} class. Each view controls a particular rectangular space within the window. Parent views contain and organize the layout of their children. Leaf views (those at the bottom of the hierarchy) draw in the rectangles they control and respond to user actions directed at that space. Thus, views are where the activity's interaction with the user takes place. For example, a view might display a small image and initiate an action when the user taps that image. Android has a number of ready-made views that you can use — including buttons, text fields, scroll bars, menu items, check boxes, and more. A view hierarchy is placed within an activity's window by the {@link android.app.Activity#setContentView Activity.setContentView()} method. The *content view* is the View object at the root of the hierarchy. (See the separate User Interface document for more information on views and the hierarchy.) **Services** A *service* doesn't have a visual user interface, but rather runs in the background for an indefinite period of time. For example, a service might play background music as the user attends to other matters, or it

'701 Claims	Android Device with One or More Apps
	might fetch data over the network or calculate something and provide the result to activities that need it. Each service extends the {@link android.app.Service} base class. A prime example is a media player playing songs from a play list. The player application would probably have one or more activities that allow the user to choose songs and start playing them. However, the music playback itself would not be handled by an activity because users will expect the music to keep playing even after they leave the player and begin something different. To keep the music going, the media player activity could start a service to run in the background. The system would then keep the music playback service running even after the activity that started it leaves the screen. It's possible to connect to (bind to) an ongoing service (and start the service if it's not already running). While connected, you can communicate with the service through an interface that the service exposes. For the music service, this interface might allow users to pause, rewind, stop, and restart the playback. Like activities and the other components, services run in the main thread of the application process. So that they won't block other components or the user interface, they often spawn another thread for time-consuming tasks (like music playback). See Processes and Threads, later. All the activities in a task move together as a unit. The entire task (the entire activity stack) can be brought to the foreground or sent to the background. Suppose, for instance, that the current task has four activities in its stack — three under the current activity. The user presses the HOME key, goes to the application launcher, and selects a new application (actually, a new *task*). The current task goes into the background and the root activity for the new task is displayed. Then, after a short period, the user goes back to the home screen and again selects the previous application (the previous task). That task, with all four activities in the stack, comes forward. When the user presses the BACK key, the screen does not display the activity the user just left (the root activity of the previous task). Rather, the activity on the top of the stack is removed and the previous activity in the same task is displayed. As noted above, there's never more than one instance of a "{@code singleTask}" or "{@code singleInstance}" activity, so that instance is expected to handle all new intents. A "{@code

'701 Claims	Android Device with One or More Apps
	singleInstance}" activity is always at the top of the stack (since it is the only activity in the task), so it is always in position to handle the intent. However, a "{@code singleTask}" activity may or may not have other activities above it in the stack. If it does, it is not in position to handle the intent, and the intent is dropped. (Even though the intent is dropped, its arrival would have caused the task to come to the foreground, where it would remain.) An activity has essentially three states: • It is *active* or *running* when it is in the foreground of the screen (at the top of the activity stack for the current task). This is the activity that is the focus for the user's actions. • It is *paused* if it has lost focus but is still visible to the user. That is, another activity lies on top of it and that activity either is transparent or doesn't cover the full screen, so some of the paused activity can show through. A paused activity is completely alive (it maintains all state and member information and remains attached to the window manager), but can be killed by the system in extreme low memory situations. • It is *stopped* if it is completely obscured by another activity. It still retains all state and member information. However, it is no longer visible to the user so its window is hidden and it will often be killed by the system when memory is needed elsewhere. Taken together, these seven methods define the entire lifecycle of an activity. There are three nested loops that you can monitor by implementing them: • The **entire lifetime** of an activity happens between the first call to {@link android.app.Activity#onCreate onCreate()} through to a single final call to {@link android.app.Activity#onDestroy}. An activity does all its initial setup of "global" state in {@code onCreate()}, and releases all remaining resources in {@code onDestroy()}. For example, if it has a thread running in the background to download data from the network, it may create that thread in {@code onCreate()} and then stop the thread in {@code onDestroy()}. • The **visible lifetime** of an activity happens between a call to {@link android.app.Activity#onStart onStart()} until a corresponding call to {@link android.app.Activity#onStop onStop()}. During this time, the user can see the activity on-screen, though it may not be in the foreground and interacting with the user. Between these two methods, you can maintain resources that are needed

<table>
<tr><th>'701 Claims</th><th>Android Device with One or More Apps</th></tr>
<tr><td></td><td>
to show the activity to the user. For example, you can register a {@link android.content.BroadcastReceiver} in {@code onStart()} to monitor for changes that impact your UI, and unregister it in {@code onStop()} when the user can no longer see what you are displaying. The {@code onStart()} and {@code onStop()} methods can be called multiple times, as the activity alternates between being visible and hidden to the user.
• The <b>foreground lifetime</b> of an activity happens between a call to {@link android.app.Activity#onResume onResume()} until a corresponding call to {@link android.app.Activity#onPause onPause()}. During this time, the activity is in front of all other activities on screen and is interacting with the user. An activity can frequently transition between the resumed and paused states — for example, {@code onPause()} is called when the device goes to sleep or when a new activity is started, {@code onResume()} is called when an activity result or a new intent is delivered. Therefore, the code in these two methods should be fairly lightweight.

<table>
<tr><td>{@link android.app.Activity#onStart onStart()}</td><td>Called just before the activity becomes visible to the user.
Followed by {@code onResume()} if the activity comes to the foreground, or {@code onStop()} if it becomes hidden.</td><td>No</td><td>{@code onResume()} or {@code onStop()}</td></tr>
</table>
Processes and lifecycles
The Android system tries to maintain an application process for as long as possible, but eventually it will need to remove old processes when memory runs low. To determine which processes to keep and which to kill, Android places each process into an "importance hierarchy" based on the components running in it and the state of those components. Processes with the lowest importance are eliminated first, then those with the next lowest, and so on. There are five levels in the hierarchy. The following list presents them in order of importance:
6. A <b>foreground process</b> is one that is required for what the user is currently doing. A process is considered to be in the foreground if any of the following conditions hold:
o It is running an activity that the user is interacting with (the Activity object's {@link android.app.Activity#onResume onResume()} method has been called).
</td></tr>
</table>

'701 Claims	Android Device with One or More Apps
	o It hosts a service that's bound to the activity that the user is interacting with. o It has a {@link android.app.Service} object that's executing one of its lifecycle callbacks ({@link android.app.Service#onCreate onCreate()}, {@link android.app.Service#onStart onStart()}, or {@link android.app.Service#onDestroy onDestroy()}). o It has a {@link android.content.BroadcastReceiver} object that's executing its {@link android.content.BroadcastReceiver#onReceive onReceive()} method. Only a few foreground processes will exist at any given time. They are killed only as a last resort — if memory is so low that they cannot all continue to run. Generally, at that point, the device has reached a memory paging state, so killing some foreground processes is required to keep the user interface responsive. 7. A **visible process** is one that doesn't have any foreground components, but still can affect what the user sees on screen. A process is considered to be visible if either of the following conditions holds: o It hosts an activity that is not in the foreground, but is still visible to the user (its {@link android.app.Activity#onPause onPause()} method has been called). This may occur, for example, if the foreground activity is a dialog that allows the previous activity to be seen behind it. o It hosts a service that's bound to a visible activity. A visible process is considered extremely important and will not be killed unless doing so is required to keep all foreground processes running. 8. A **service process** is one that is running a service that has been started with the {@link android.content.Context#startService startService()} method and that does not fall into either of the two higher categories. Although service processes are not directly tied to anything the user sees, they are generally doing things that the user cares about (such as playing an mp3 in the background or downloading data on the network), so the system keeps them running unless there's not enough memory to retain them along with all foreground and visible processes. 9. A **background process** is one holding an activity that's not currently visible to the user (the Activity object's {@link android.app.Activity#onStop onStop()} method has been called). These processes have no direct impact on the user experience, and can be killed at any time to reclaim memory for a foreground, visible, or service process. Usually there are many background

'701 Claims	Android Device with One or More Apps
	processes running, so they are kept in an LRU (least recently used) list to ensure that the process with the activity that was most recently seen by the user is the last to be killed. If an activity implements its lifecycle methods correctly, and captures its current state, killing its process will not have a deleterious effect on the user experience. 10. An **empty process** is one that doesn't hold any active application components. The only reason to keep such a process around is as a cache to improve startup time the next time a component needs to run in it. The system often kills these processes in order to balance overall system resources between process caches and the underlying kernel caches. **SAMSUNG_PRIORART0005487, Activity** * <p>Activities in the system are managed as an <em>activity stack</em>. * When a new activity is started, it is placed on the top of the stack * and becomes the running activity -- the previous activity always remains * below it in the stack, and will not come to the foreground again until * the new activity exits.</p> * * <p>An activity has essentially four states:</p> * <ul> * <li> If an activity in the foreground of the screen (at the top of * the stack), * it is <em>active</em> or <em>running</em>. </li> * <li>If an activity has lost focus but is still visible (that is, a new non-full-sized * or transparent activity has focus on top of your activity), it * is <em>paused</em>. A paused activity is completely alive (it * maintains all state and member information and remains attached to * the window manager), but can be killed by the system in extreme * low memory situations.

'701 Claims	Android Device with One or More Apps
	* <li>If an activity is completely obscured by another activity, * it is <em>stopped</em>. It still retains all state and member information, * however, it is no longer visible to the user so its window is hidden * and it will often be killed by the system when memory is needed * elsewhere.</li> * <li>If an activity is paused or stopped, the system can drop the activity * from memory by either asking it to finish, or simply killing its * process. When it is displayed again to the user, it must be * completely restarted and restored to its previous state.</li> * </ul> * * <p>The following diagram shows the important state paths of an Activity. * The square rectangles represent callback methods you can implement to * perform operations when the Activity moves between states. The colored * ovals are major states the Activity can be in.</p> * * <p><img src="../../../images/activity_lifecycle.png" * alt="State diagram for an Android Activity Lifecycle." border="0" /></p> * * <p>There are three key loops you may be interested in monitoring within your * activity: * * <ul> * <li>The <b>entire lifetime</b> of an activity happens between the first call * to {@link android.app.Activity#onCreate} through to a single final call * to {@link android.app.Activity#onDestroy}. An activity will do all setup * of "global" state in onCreate(), and release all remaining resources in * onDestroy(). For example, if it has a thread running in the background * to download data from the network, it may create that thread in onCreate() * and then stop the thread in onDestroy().

'701 Claims	Android Device with One or More Apps
	* * <li>The <b>visible lifetime</b> of an activity happens between a call to * {@link android.app.Activity#onStart} until a corresponding call to * {@link android.app.Activity#onStop}. During this time the user can see the * activity on-screen, though it may not be in the foreground and interacting * with the user. Between these two methods you can maintain resources that * are needed to show the activity to the user. For example, you can register * a {@link android.content.BroadcastReceiver} in onStart() to monitor for changes * that impact your UI, and unregister it in onStop() when the user an no * longer see what you are displaying. The onStart() and onStop() methods * can be called multiple times, as the activity becomes visible and hidden * to the user. * * <li>The <b>foreground lifetime</b> of an activity happens between a call to * {@link android.app.Activity#onResume} until a corresponding call to * {@link android.app.Activity#onPause}. During this time the activity is * in front of all other activities and interacting with the user. An activity * can frequently go between the resumed and paused states -- for example when * the device goes to sleep, when an activity result is delivered, when a new * intent is delivered -- so the code in these methods should be fairly * lightweight. * </ul> * * <tr><th colspan="2" align="left" border="0">{@link android.app.Activity#onStart onStart()}</th> * <td>Called when the activity is becoming visible to the user. * <p>Followed by <code>onResume()</code> if the activity comes * to the foreground, or <code>onStop()</code> if it becomes hidden.</td> * <td align="center">No</td>

'701 Claims	Android Device with One or More Apps
	* &nbsp;&nbsp;&nbsp;&nbsp; <td align="center"><code>onResume()</code> or <code>onStop()</code></td> * &nbsp;&nbsp;&nbsp;&nbsp;</tr> * * &nbsp; <tr><td rowspan="2" style="border-left: none;">&nbsp;&nbsp;&nbsp;&nbsp;</td> * &nbsp;&nbsp;&nbsp; <th align="left" border="0">{@link android.app.Activity#onResume onResume()}</th> * &nbsp;&nbsp;&nbsp; <td>Called when the activity will start * &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp; interacting with the user. At this point your activity is at * &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp; the top of the activity stack, with user input going to it. * &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp; <p>Always followed by <code>onPause()</code>.</td> * &nbsp;&nbsp;&nbsp; <td align="center">No</td> * &nbsp;&nbsp;&nbsp; <td align="center"><code>onPause()</code></td> * &nbsp;&nbsp;&nbsp;&nbsp;</tr> * * * <p>Unless you specify otherwise, a configuration change (such as a change * in screen orientation, language, input devices, etc) will cause your * current activity to be <em>destroyed</em>, going through the normal activity * lifecycle process of {@link #onPause}, * {@link #onStop}, and {@link #onDestroy} as appropriate. If the activity * had been in the foreground or visible to the user, once {@link #onDestroy} is * called in that instance then a new instance of the activity will be * created, with whatever savedInstanceState the previous instance had generated * from {@link #onSaveInstanceState}.</p> * <li> <p>The <b>foreground activity</b> (the activity at the top of the screen * that the user is currently interacting with) is considered the most important. * Its process will only be killed as a last resort, if it uses more memory * than is available on the device. Generally at this point the device has * reached a memory paging state, so this is required in order to keep the user

'701 Claims	Android Device with One or More Apps
	* interface responsive. * <li> <p>A <b>visible activity</b> (an activity that is visible to the user * but not in the foreground, such as one sitting behind a foreground dialog) * is considered extremely important and will not be killed unless that is * required to keep the foreground activity running. * <li> <p>A <b>background activity</b> (an activity that is not visible to * the user and has been paused) is no longer critical, so the system may * safely kill its process to reclaim memory for other foreground or * visible processes. If its process needs to be killed, when the user navigates * back to the activity (making it visible on the screen again), its * {@link #onCreate} method will be called with the savedInstanceState it had previously * supplied in {@link #onSaveInstanceState} so that it can restart itself in the same * state as the user last left it. * <li> <p>An <b>empty process</b> is one hosting no activities or other * application components (such as {@link Service} or * {@link android.content.BroadcastReceiver} classes). These are killed very * quickly by the system as memory becomes low. For this reason, any * background operation you do outside of an activity must be executed in the * context of an activity BroadcastReceiver or Service to ensure that the system * knows it needs to keep your process around. * </ol> * * <p>Sometimes an Activity may need to do a long-running operation that exists * independently of the activity lifecycle itself. An example may be a camera * application that allows you to upload a picture to a web site. The upload * may take a long time, and the application should allow the user to leave * the application will it is executing. To accomplish this, your Activity * should start a {@link Service} in which the upload takes place. This allows * the system to properly prioritize your process (considering it to be more * important than other non-visible applications) for the duration of the

’701 Claims	Android Device with One or More Apps

	* upload, independent of whether the original activity is paused, stopped,
	* or finished.
	*/
	
	/**
	* Called as part of the activity lifecycle when an activity is about to go
	* into the background as the result of user choice. For example, when the
	* user presses the Home key, {@link #onUserLeaveHint} will be called, but
	* when an incoming phone call causes the in-call Activity to be automatically
	* brought to the foreground, {@link #onUserLeaveHint} will not be called on
	* the activity being interrupted. In cases when it is invoked, this method
	* is called right before the activity's {@link #onPause} callback.
	*
	* <p>This callback and {@link #onUserInteraction} are intended to help
	* activities manage status bar notifications intelligently; specifically,
	* for helping activities determine the proper time to cancel a notfication.
	*
	* @see #onUserInteraction()
	*/
	protected void onUserLeaveHint() {
	}
	
	* <p>As a general rule, however, a resumed activity will have window
	* focus... unless it has displayed other dialogs or popups that take
	* input focus, in which case the activity itself will not have focus
	* when the other windows have it. Likewise, the system may display
	* system-level windows (such as the status bar notification panel or
	* a system alert) which will temporarily take window input focus without
	* pausing the foreground activity.
	*

'701 Claims	Android Device with One or More Apps
	* @param hasFocus Whether the window of this activity has focus. * * @see #hasWindowFocus() /** * Change the desired orientation of this activity. If the activity * is currently in the foreground or otherwise impacting the screen * orientation, the screen will immediately be changed (possibly causing * the activity to be restarted). Otherwise, this will be used the next * time the activity is visible. * * @param requestedOrientation An orientation constant as used in * {@link ActivityInfo#screenOrientation ActivityInfo.screenOrientation}. */ public void setRequestedOrientation(int requestedOrientation) { **GOOG-HEADWATER-00000029, SAMSUNG_PRIORART0005353, ConnectivityManager** /** * Class that answers queries about the state of network connectivity. It also * notifies applications when network connectivity changes. Get an instance * of this class by calling * {@link android.content.Context#getSystemService(String) Context.getSystemService(Context.CONNECTIVITY_SERVICE)}. * <p> * The primary responsibilities of this class are to: * <ol> * <li>Monitor network connections (Wi-Fi, GPRS, UMTS, etc.)</li>

'701 Claims	Android Device with One or More Apps
	* <li>Send broadcast intents when network connectivity changes</li> * <li>Attempt to "fail over" to another network when connectivity to a network * is lost</li> * <li>Provide an API that allows applications to query the coarse-grained or fine-grained * state of the available networks</li> * </ol> */ * A change in network connectivity has occurred. A connection has either * been established or lost. The NetworkInfo for the affected network is * sent as an extra; it should be consulted to see what kind of * connectivity event occurred. /** * Broadcast Action: The setting for **background data usage** has changed * values. Use {@link #getBackgroundDataSetting()} to get the current value. * <p> * If an application uses the network in the background, it should listen * for this broadcast and stop using the background data if the value is * false. */ @SdkConstant(SdkConstantType.BROADCAST_INTENT_ACTION) public static final String ACTION_BACKGROUND_DATA_SETTING_CHANGED = "android.net.conn.BACKGROUND_DATA_SETTING_CHANGED"; /** * The Default Mobile data connection. When active, all data traffic * will use this connection by default. Should not coexist with other * default connections.

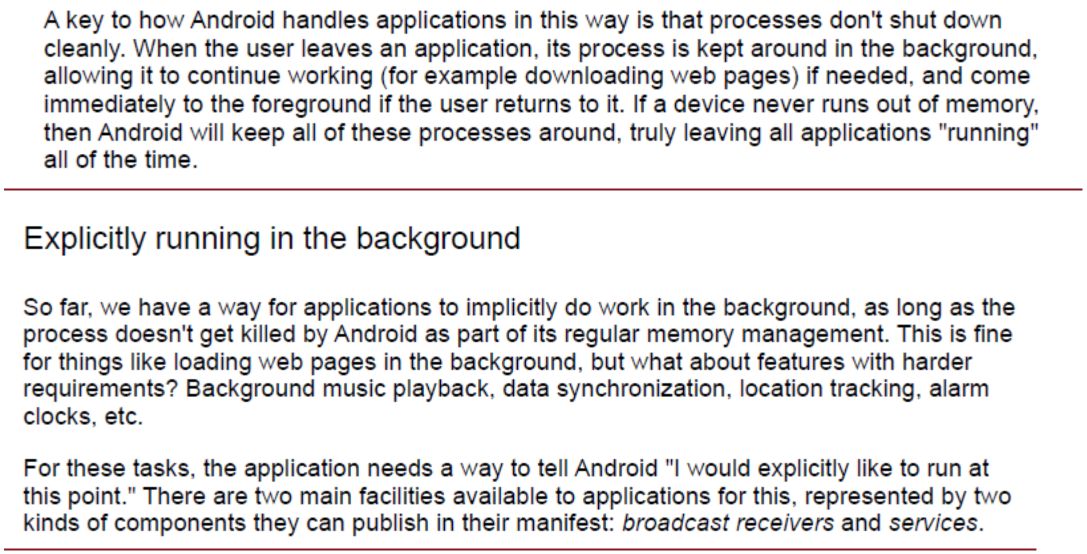
'701 Claims	Android Device with One or More Apps

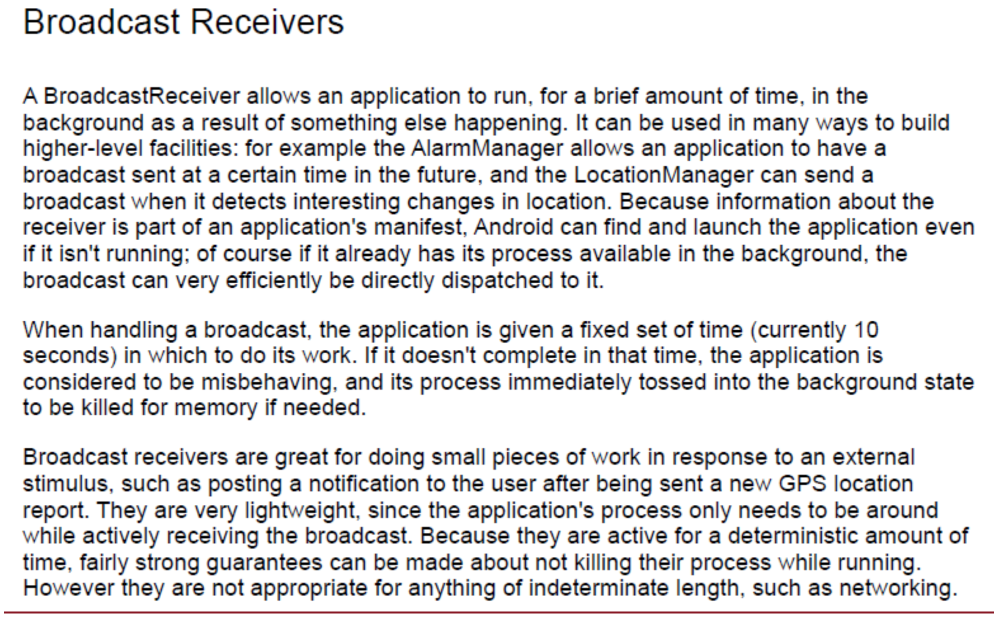
	*/
	public static final int TYPE_MOBILE = 0;
	/**
	* The Default WIFI data connection. When active, all data traffic
	* will use this connection by default. Should not coexist with other
	* default connections.
	*/
	public static final int TYPE_WIFI = 1;
	
	/**
	* Returns the value of the setting **for background data usage**. If false,
	* applications should not use the network if the application is not in the
	* foreground. Developers should respect this setting, and check the value
	* of this before performing any background data operations.
	* <p>
	* All applications that have background services that use the network
	* should listen to {@link #ACTION_BACKGROUND_DATA_SETTING_CHANGED}.
	*
	* @return Whether background data usage is allowed.
	*/
	public boolean getBackgroundDataSetting() {
	try {
	return mService.getBackgroundDataSetting();
	} catch (RemoteException e) {
	// Err on the side of safety
	return false;
	}
	}
	/**

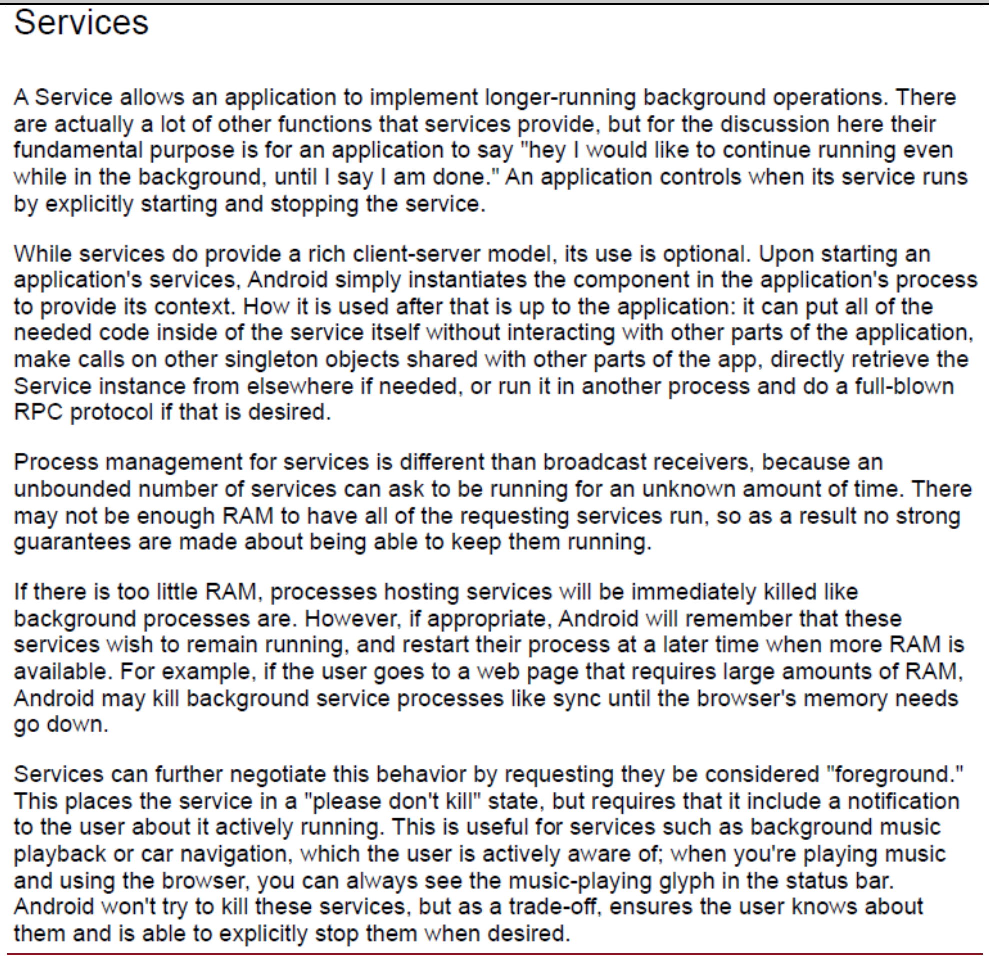
<table>
<tr><th>’701 Claims</th><th>Android Device with One or More Apps</th></tr>
<tr><td></td><td>

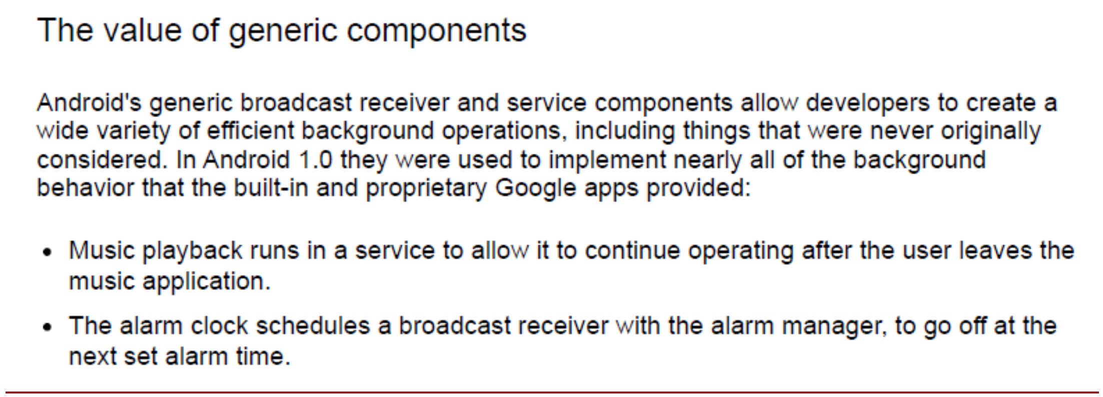
```
    * Sets the value of the setting for background data usage.
    *
    * @param allowBackgroundData Whether an application should use data while
    *            it is in the background.
    *
    * @attr ref android.Manifest.permission#CHANGE_BACKGROUND_DATA_SETTING
    * @see #getBackgroundDataSetting()
    * @hide
    */
   public void setBackgroundDataSetting(boolean allowBackgroundData) {
       try {
           mService.setBackgroundDataSetting(allowBackgroundData);
       } catch (RemoteException e) {
       }
   }



   /**
    * Sets the persisted value for enabling/disabling Mobile data.
    *
    * @param enabled Whether the mobile data connection should be
    *            used or not.
    * @hide
    */
   public void setMobileDataEnabled(boolean enabled) {
       try {
           mService.setMobileDataEnabled(enabled);
       } catch (RemoteException e) {
       }
   }
```

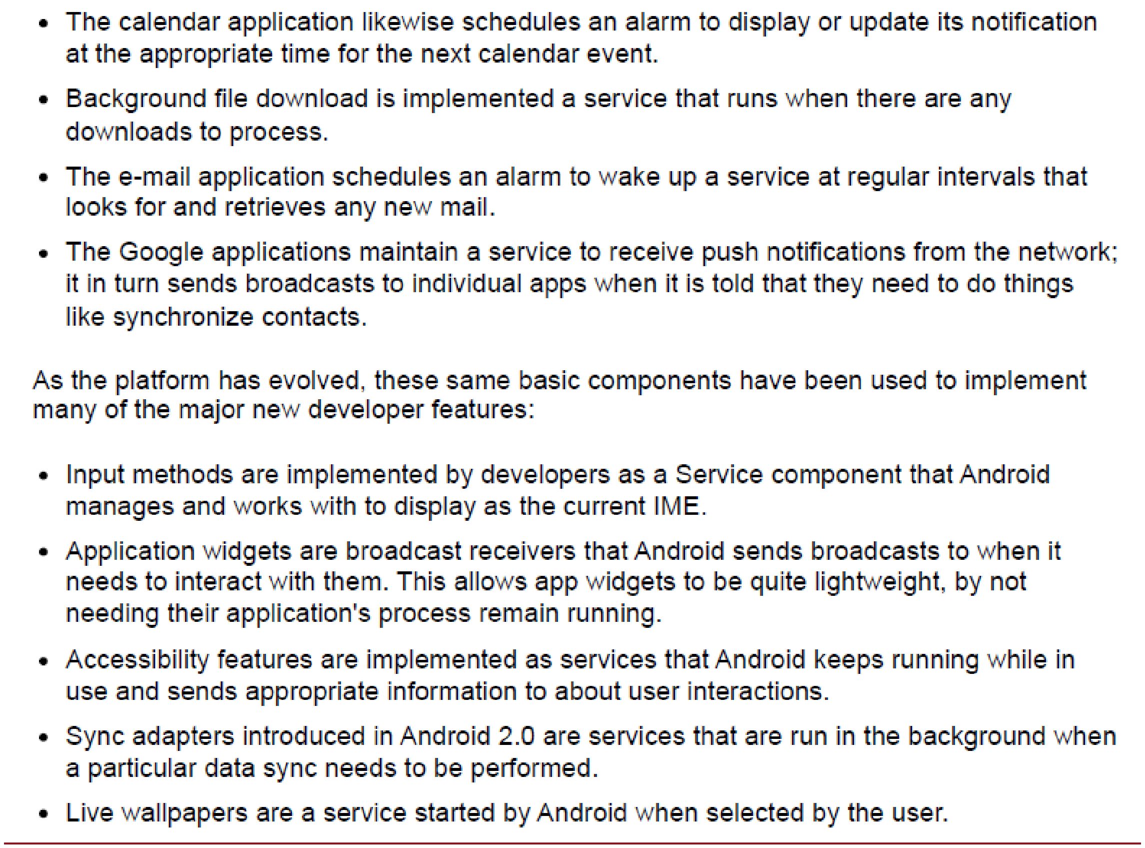
</td></tr>
</table>

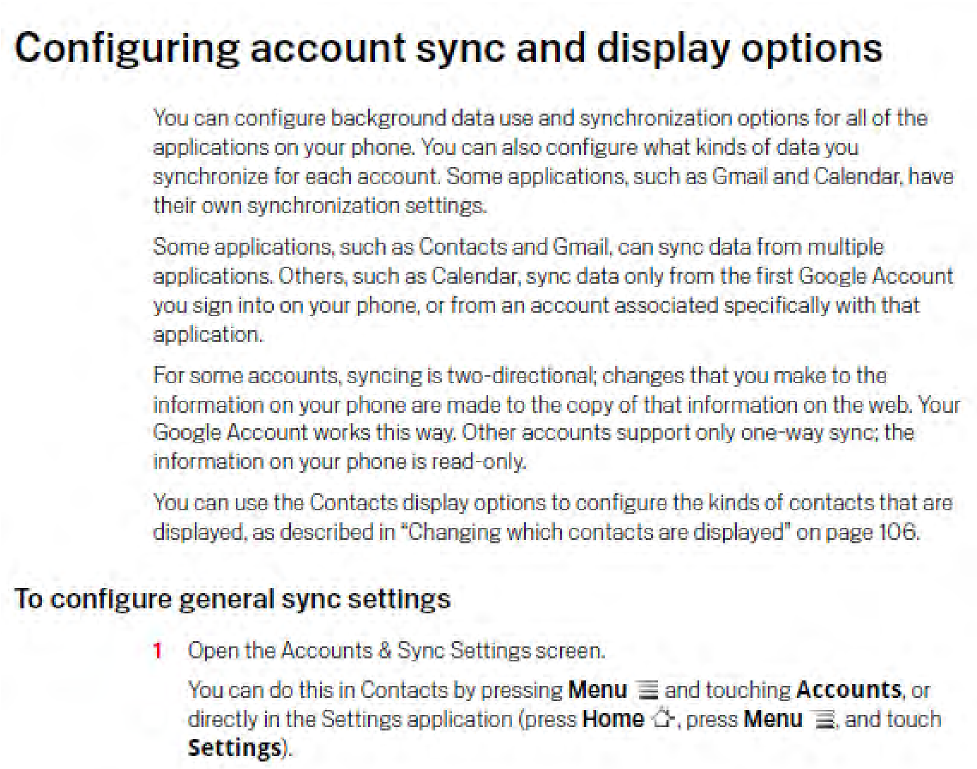
’701 Claims	Android Device with One or More Apps
	See also Android Developers Blog Multitasking the Android Way, GOOG-HEADWATER-00000025-27  A key to how Android handles applications in this way is that processes don't shut down cleanly. When the user leaves an application, its process is kept around in the background, allowing it to continue working (for example downloading web pages) if needed, and come immediately to the foreground if the user returns to it. If a device never runs out of memory, then Android will keep all of these processes around, truly leaving all applications "running" all of the time. Explicitly running in the background So far, we have a way for applications to implicitly do work in the background, as long as the process doesn't get killed by Android as part of its regular memory management. This is fine for things like loading web pages in the background, but what about features with harder requirements? Background music playback, data synchronization, location tracking, alarm clocks, etc. For these tasks, the application needs a way to tell Android "I would explicitly like to run at this point." There are two main facilities available to applications for this, represented by two kinds of components they can publish in their manifest: *broadcast receivers* and *services*.

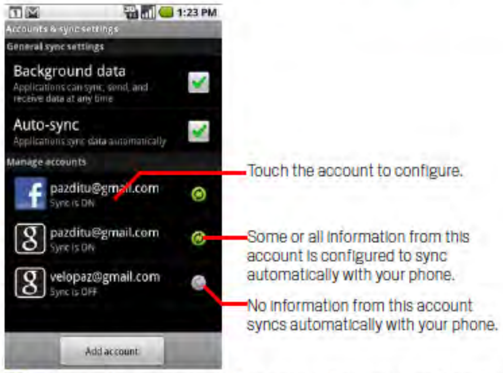
'701 Claims	Android Device with One or More Apps
	 Broadcast Receivers A BroadcastReceiver allows an application to run, for a brief amount of time, in the background as a result of something else happening. It can be used in many ways to build higher-level facilities: for example the AlarmManager allows an application to have a broadcast sent at a certain time in the future, and the LocationManager can send a broadcast when it detects interesting changes in location. Because information about the receiver is part of an application's manifest, Android can find and launch the application even if it isn't running; of course if it already has its process available in the background, the broadcast can very efficiently be directly dispatched to it. When handling a broadcast, the application is given a fixed set of time (currently 10 seconds) in which to do its work. If it doesn't complete in that time, the application is considered to be misbehaving, and its process immediately tossed into the background state to be killed for memory if needed. Broadcast receivers are great for doing small pieces of work in response to an external stimulus, such as posting a notification to the user after being sent a new GPS location report. They are very lightweight, since the application's process only needs to be around while actively receiving the broadcast. Because they are active for a deterministic amount of time, fairly strong guarantees can be made about not killing their process while running. However they are not appropriate for anything of indeterminate length, such as networking.

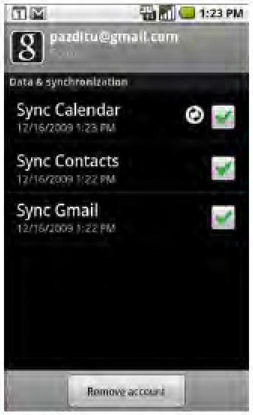
’701 Claims	Android Device with One or More Apps
	 Services A Service allows an application to implement longer-running background operations. There are actually a lot of other functions that services provide, but for the discussion here their fundamental purpose is for an application to say "hey I would like to continue running even while in the background, until I say I am done." An application controls when its service runs by explicitly starting and stopping the service. While services do provide a rich client-server model, its use is optional. Upon starting an application's services, Android simply instantiates the component in the application's process to provide its context. How it is used after that is up to the application: it can put all of the needed code inside of the service itself without interacting with other parts of the application, make calls on other singleton objects shared with other parts of the app, directly retrieve the Service instance from elsewhere if needed, or run it in another process and do a full-blown RPC protocol if that is desired. Process management for services is different than broadcast receivers, because an unbounded number of services can ask to be running for an unknown amount of time. There may not be enough RAM to have all of the requesting services run, so as a result no strong guarantees are made about being able to keep them running. If there is too little RAM, processes hosting services will be immediately killed like background processes are. However, if appropriate, Android will remember that these services wish to remain running, and restart their process at a later time when more RAM is available. For example, if the user goes to a web page that requires large amounts of RAM, Android may kill background service processes like sync until the browser's memory needs go down. Services can further negotiate this behavior by requesting they be considered "foreground." This places the service in a "please don't kill" state, but requires that it include a notification to the user about it actively running. This is useful for services such as background music playback or car navigation, which the user is actively aware of; when you're playing music and using the browser, you can always see the music-playing glyph in the status bar. Android won't try to kill these services, but as a trade-off, ensures the user knows about them and is able to explicitly stop them when desired.

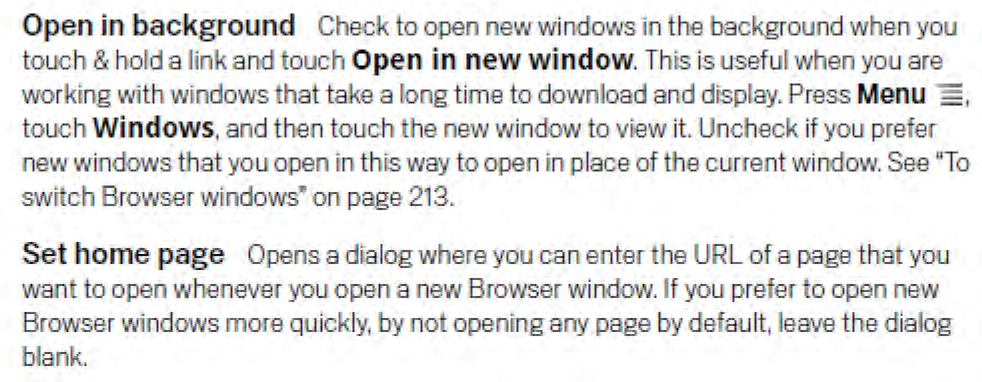
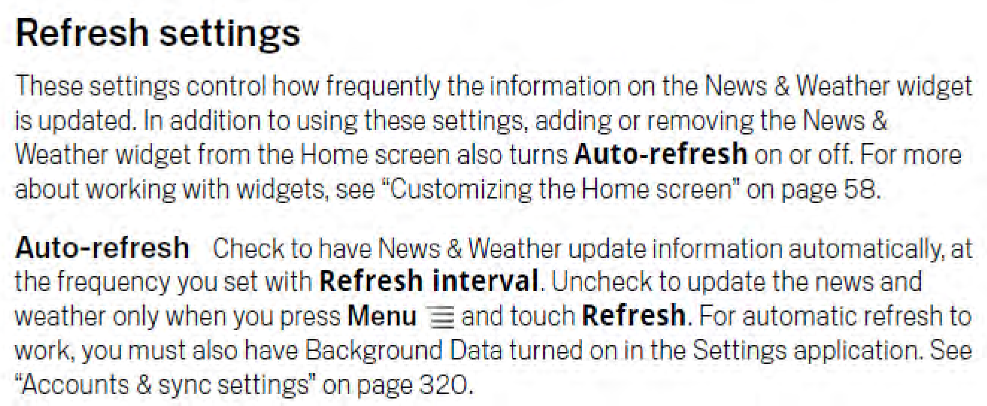
'701 Claims	Android Device with One or More Apps
	 The value of generic components Android's generic broadcast receiver and service components allow developers to create a wide variety of efficient background operations, including things that were never originally considered. In Android 1.0 they were used to implement nearly all of the background behavior that the built-in and proprietary Google apps provided: • Music playback runs in a service to allow it to continue operating after the user leaves the music application. • The alarm clock schedules a broadcast receiver with the alarm manager, to go off at the next set alarm time.

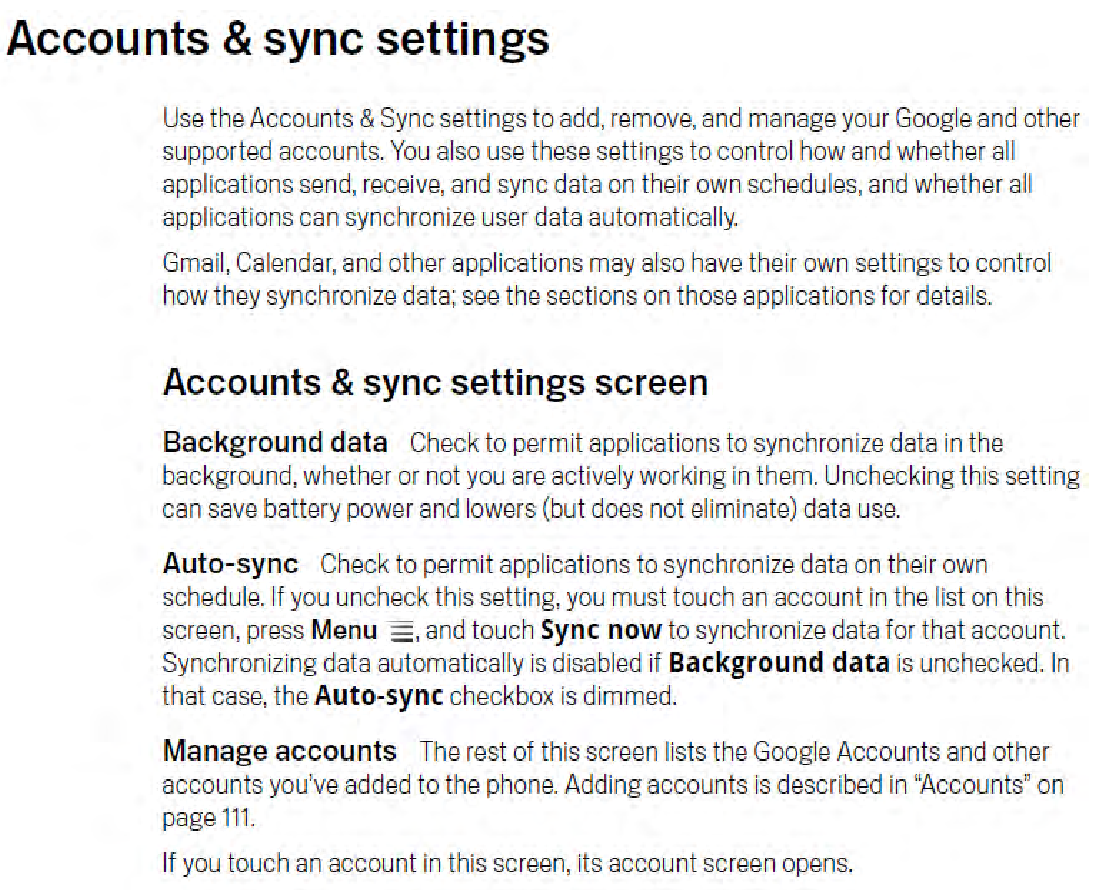
'701 Claims	Android Device with One or More Apps
	 • The calendar application likewise schedules an alarm to display or update its notification at the appropriate time for the next calendar event. • Background file download is implemented a service that runs when there are any downloads to process. • The e-mail application schedules an alarm to wake up a service at regular intervals that looks for and retrieves any new mail. • The Google applications maintain a service to receive push notifications from the network; it in turn sends broadcasts to individual apps when it is told that they need to do things like synchronize contacts. As the platform has evolved, these same basic components have been used to implement many of the major new developer features: • Input methods are implemented by developers as a Service component that Android manages and works with to display as the current IME. • Application widgets are broadcast receivers that Android sends broadcasts to when it needs to interact with them. This allows app widgets to be quite lightweight, by not needing their application's process remain running. • Accessibility features are implemented as services that Android keeps running while in use and sends appropriate information to about user interactions. • Sync adapters introduced in Android 2.0 are services that are run in the background when a particular data sync needs to be performed. • Live wallpapers are a service started by Android when selected by the user.
[1e] wherein the access is selectively blocked based on a determination that the first end-user	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for ~~claims~~ [1d]. In addition:

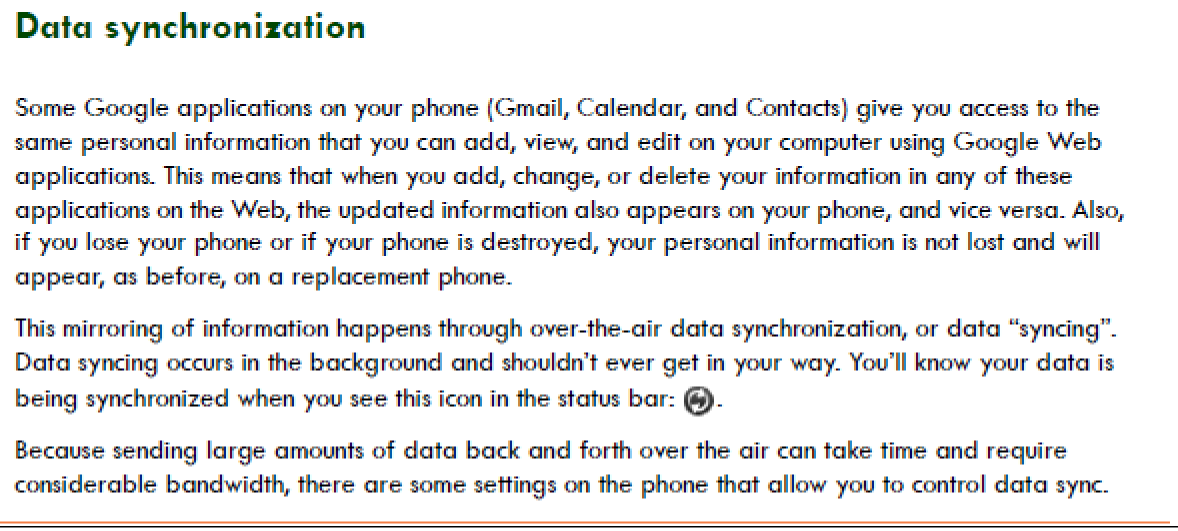
'701 Claims	Android Device with One or More Apps
application is running in a background state, and wherein the access is selectively allowed based on a determination that the first end-user application is running as a foreground application.	**Nexus One** *See, e.g.*, SAMSUNG_PRIORART0000001 (Nexus) at 115-117:  **Configuring account sync and display options** You can configure background data use and synchronization options for all of the applications on your phone. You can also configure what kinds of data you synchronize for each account. Some applications, such as Gmail and Calendar, have their own synchronization settings. Some applications, such as Contacts and Gmail, can sync data from multiple applications. Others, such as Calendar, sync data only from the first Google Account you sign into on your phone, or from an account associated specifically with that application. For some accounts, syncing is two-directional; changes that you make to the information on your phone are made to the copy of that information on the web. Your Google Account works this way. Other accounts support only one-way sync; the information on your phone is read-only. You can use the Contacts display options to configure the kinds of contacts that are displayed, as described in "Changing which contacts are displayed" on page 106. **To configure general sync settings** 1 Open the Accounts & Sync Settings screen. You can do this in Contacts by pressing **Menu** ☰ and touching **Accounts**, or directly in the Settings application (press **Home** ⌂, press **Menu** ☰, and touch **Settings**).

'701 Claims	Android Device with One or More Apps
	The screen displays your current sync settings and a list of your current accounts.  1:23 PM Accounts & sync settings General sync settings Background data — Applications can sync, send, and receive data at any time ☑ Auto-sync — Applications sync data automatically ☑ Manage accounts pazditu@gmail.com — Sync is ON pazditu@gmail.com — Sync is ON velopaz@gmail.com — Sync is OFF Add account Touch the account to configure. Some or all information from this account is configured to sync automatically with your phone. No information from this account syncs automatically with your phone. ● indicates that some or all of an account's information is configured to sync automatically with your phone. ○ indicates that none of an account's information is configured to sync automatically with your phone. 2 Check or uncheck **Background data** to control whether applications and services can transmit data when you are not working with them directly (that is, when they are running in the background). If you uncheck this option, Gmail stops receiving new mail, Calendar stops syncing events, and so on, until you touch the **Refresh** menu item or send an email. 3 Check or uncheck **Auto-sync** to control whether changes you make to information on the phone or on the web are automatically synced with each other. For example, when this option is checked, changes that you make in Contacts on the phone are automatically made in Google Contacts on the web. If you uncheck this option, you may be able to use an application's tools to sync data manually. See "To sync information manually" on page 117.

'701 Claims	Android Device with One or More Apps
	To sync information manually 1 Open the Accounts & Sync Settings screen. 2 Touch the account whose data you want to sync. 3 Press **Menu** ☰ and touch **Sync now**. **To change an account's sync settings** 1 Open the Accounts & Sync Settings screen. 2 Touch the account whose sync settings you want to change. The Data and Synchronization screen opens, displaying a list of the kinds of information the account can sync. Checked items are configured to sync to your phone.  1:23 PM pazditu@gmail.com Data & synchronization Sync Calendar ☑ 12/16/2009 1:23 PM Sync Contacts ☑ 12/16/2009 1:22 PM Sync Gmail ☑ 12/16/2009 1:22 PM 3 Check or uncheck the kinds of information you want to sync to the phone. Unchecking an option does not remove the information from your phone; it simply stops it from syncing automatically. To remove the information previously synced for the account, you must remove the account. *See, e.g.*, SAMSUNG_PRIORART0000001 (Nexus) at 218:

'701 Claims	Android Device with One or More Apps
	 **Open in background** Check to open new windows in the background when you touch & hold a link and touch **Open in new window**. This is useful when you are working with windows that take a long time to download and display. Press **Menu** ☰, touch **Windows**, and then touch the new window to view it. Uncheck if you prefer new windows that you open in this way to open in place of the current window. See "To switch Browser windows" on page 213. **Set home page** Opens a dialog where you can enter the URL of a page that you want to open whenever you open a new Browser window. If you prefer to open new Browser windows more quickly, by not opening any page by default, leave the dialog blank. *See, e.g.*, SAMSUNG_PRIORART0000001 (Nexus) at 286:  **Refresh settings** These settings control how frequently the information on the News & Weather widget is updated. In addition to using these settings, adding or removing the News & Weather widget from the Home screen also turns **Auto-refresh** on or off. For more about working with widgets, see "Customizing the Home screen" on page 58. **Auto-refresh** Check to have News & Weather update information automatically, at the frequency you set with **Refresh interval**. Uncheck to update the news and weather only when you press **Menu** ☰ and touch **Refresh**. For automatic refresh to work, you must also have Background Data turned on in the Settings application. See "Accounts & sync settings" on page 320. *See, e.g.*, SAMSUNG_PRIORART0000001 (Nexus) at 320:

'701 Claims	Android Device with One or More Apps
	 **Accounts & sync settings** Use the Accounts & Sync settings to add, remove, and manage your Google and other supported accounts. You also use these settings to control how and whether all applications send, receive, and sync data on their own schedules, and whether all applications can synchronize user data automatically. Gmail, Calendar, and other applications may also have their own settings to control how they synchronize data; see the sections on those applications for details. **Accounts & sync settings screen** **Background data** Check to permit applications to synchronize data in the background, whether or not you are actively working in them. Unchecking this setting can save battery power and lowers (but does not eliminate) data use. **Auto-sync** Check to permit applications to synchronize data on their own schedule. If you uncheck this setting, you must touch an account in the list on this screen, press **Menu** ☰, and touch **Sync now** to synchronize data for that account. Synchronizing data automatically is disabled if **Background data** is unchecked. In that case, the **Auto-sync** checkbox is dimmed. **Manage accounts** The rest of this screen lists the Google Accounts and other accounts you've added to the phone. Adding accounts is described in "Accounts" on page 111. If you touch an account in this screen, its account screen opens. **HTC Dream / T-Mobile G1** SAMSUNG_PRIORART0005207-09

'701 Claims	Android Device with One or More Apps
	 **Data synchronization** Some Google applications on your phone (Gmail, Calendar, and Contacts) give you access to the same personal information that you can add, view, and edit on your computer using Google Web applications. This means that when you add, change, or delete your information in any of these applications on the Web, the updated information also appears on your phone, and vice versa. Also, if you lose your phone or if your phone is destroyed, your personal information is not lost and will appear, as before, on a replacement phone. This mirroring of information happens through over-the-air data synchronization, or data "syncing". Data syncing occurs in the background and shouldn't ever get in your way. You'll know your data is being synchronized when you see this icon in the status bar: ⟳. Because sending large amounts of data back and forth over the air can take time and require considerable bandwidth, there are some settings on the phone that allow you to control data sync.

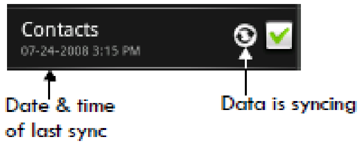
Sync by application

To control synchronization for Gmail, Calendar, and Contacts, do the following:

1 Press HOME, then press MENU and select Settings.

2 Select Data synchronization.

When any of the applications are synchronizing, you will see the "sync" icon: ⟳. You will also see the last time your data was synchronized.



3 By default, the personal information in Gmail, Calendar, and Contacts will sync whenever you make a change or receive a new message. You can change this behavior:

Auto-sync When selected, Auto-sync will sync Gmail, Calendar, and Contacts automatically, as you make changes or receive new messages. When OFF, information will not be synced automatically, although you can force a sync by using the individual application check boxes described below.

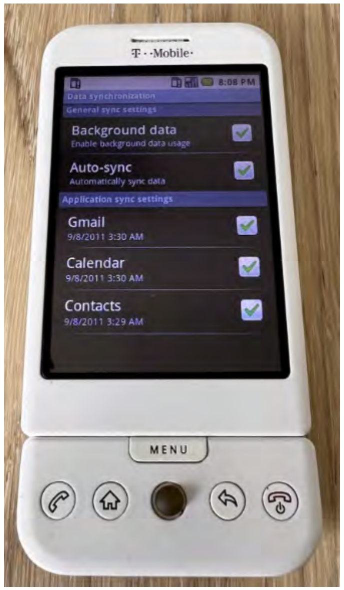
Gmail Clear this check box to exclude Gmail from auto-sync. To force a sync, either clear the check box then select it, or press MENU and select Sync now. To control sync by Gmail label, you must do so from the Gmail settings screen. Read more in "Select labels to synchronize" on page 46.

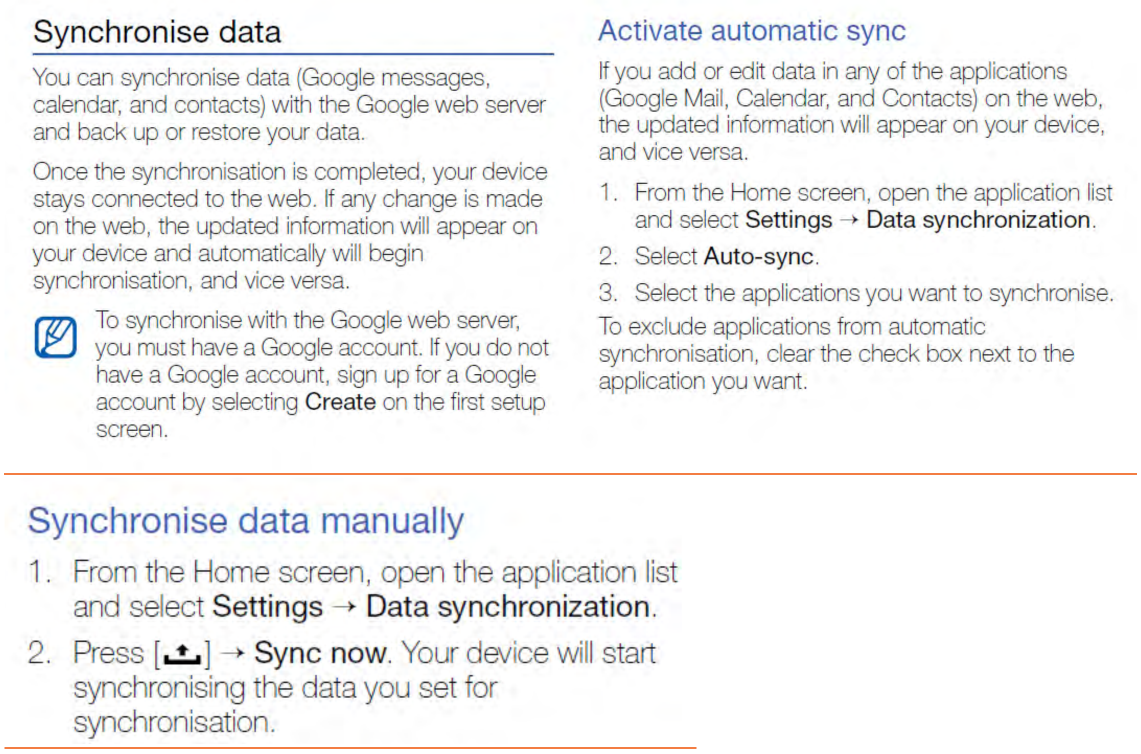
Calendar Clear this check box to exclude Calendar from auto-sync. To force a sync, either clear the check box then select it, or press MENU and select Sync now.

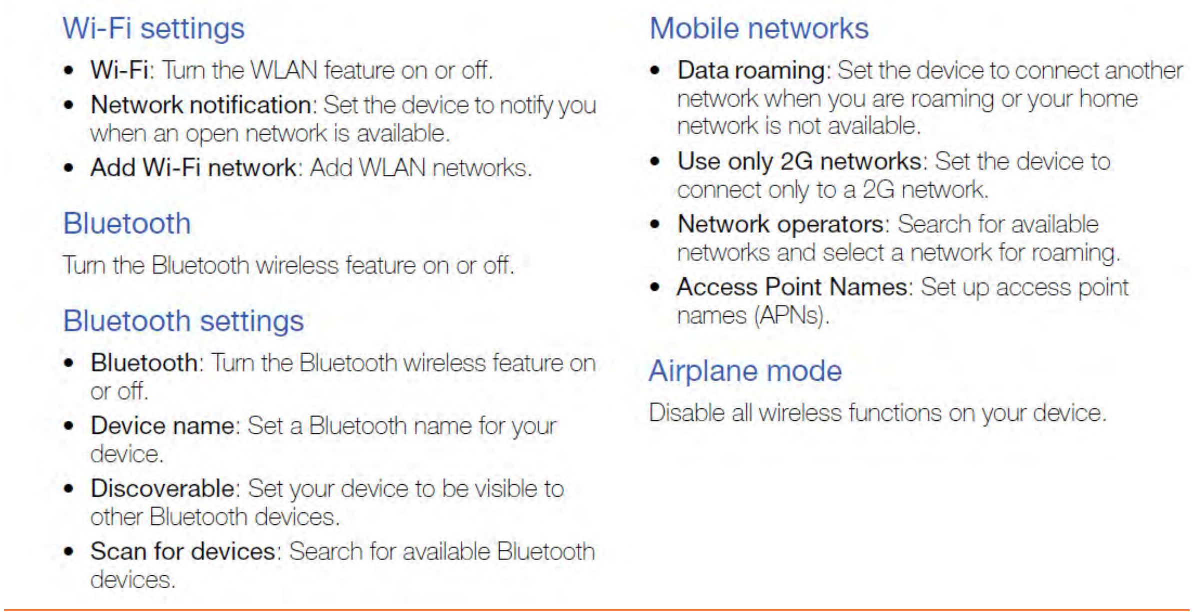
Contacts Clear this check box to exclude Contacts from auto-sync. To force a sync, either clear the check box then select it, or press MENU and select Sync now.

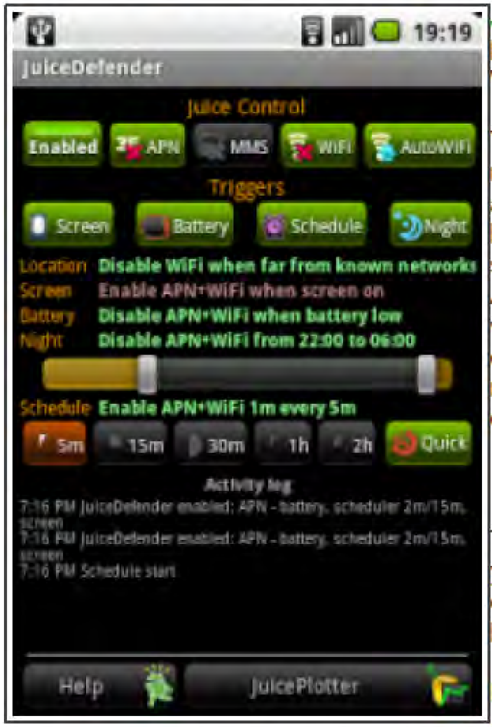
Cancel sync During a sync you can stop it by pressing MENU and selecting Cancel sync.

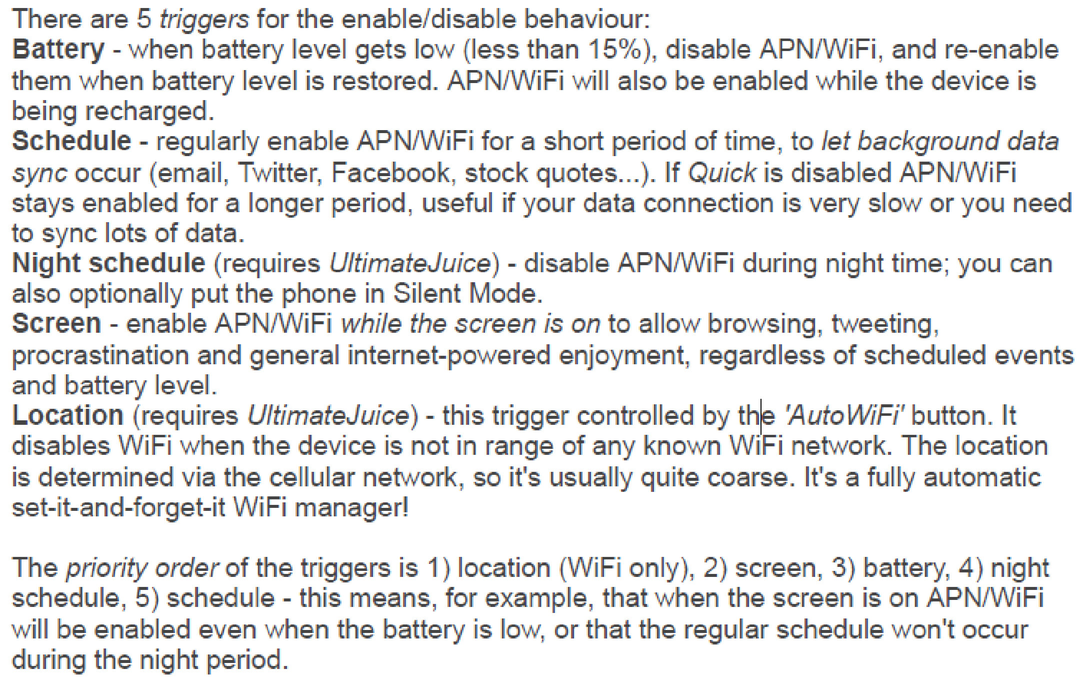
Sync problems If you see this icon ⟳ to the left of the sync check box, then there was a temporary problem with the data synchronization. Check your data connection and try again later.

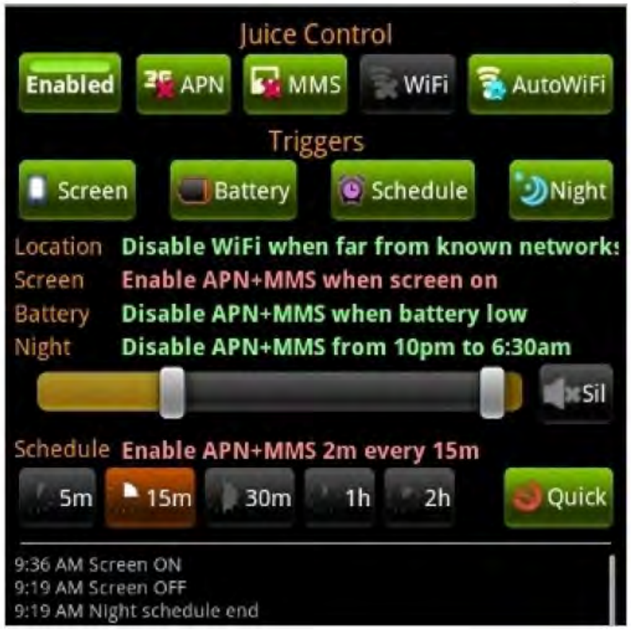
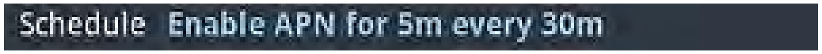
'701 Claims	Android Device with One or More Apps
	Sync by Gmail label You can select to sync only Gmail messages with certain labels from the Gmail application settings screen. Read more in "Select labels to synchronize" on page 46. SAMSUNG_PRIORART0005245 Open in background Select if you want links to new pages to open in a new window in the background. SAMSUNG_PRIORART0005598  T··Mobile· 8:08 PM Data synchronization General sync settings Background data Enable background data usage Auto-sync Automatically sync data Application sync settings Gmail 9/8/2011 3:30 AM Calendar 9/8/2011 3:30 AM Contacts 9/8/2011 3:29 AM MENU

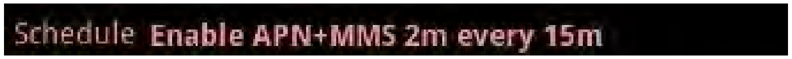
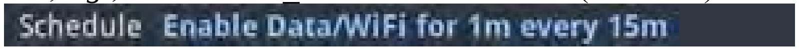
'701 Claims	Android Device with One or More Apps
	Samsung GT-I7500 Galaxy SAMSUNG_PRIORART0005494 at PDF pgs. 38-39, 72:  Synchronise data You can synchronise data (Google messages, calendar, and contacts) with the Google web server and back up or restore your data. Once the synchronisation is completed, your device stays connected to the web. If any change is made on the web, the updated information will appear on your device and automatically will begin synchronisation, and vice versa. To synchronise with the Google web server, you must have a Google account. If you do not have a Google account, sign up for a Google account by selecting **Create** on the first setup screen. Activate automatic sync If you add or edit data in any of the applications (Google Mail, Calendar, and Contacts) on the web, the updated information will appear on your device, and vice versa. 1. From the Home screen, open the application list and select **Settings** → **Data synchronization**. 2. Select **Auto-sync**. 3. Select the applications you want to synchronise. To exclude applications from automatic synchronisation, clear the check box next to the application you want. Synchronise data manually 1. From the Home screen, open the application list and select **Settings** → **Data synchronization**. 2. Press [⏏] → **Sync now**. Your device will start synchronising the data you set for synchronisation.

'701 Claims	Android Device with One or More Apps
	 Wi-Fi settings • **Wi-Fi**: Turn the WLAN feature on or off. • **Network notification**: Set the device to notify you when an open network is available. • **Add Wi-Fi network**: Add WLAN networks. Bluetooth Turn the Bluetooth wireless feature on or off. Bluetooth settings • **Bluetooth**: Turn the Bluetooth wireless feature on or off. • **Device name**: Set a Bluetooth name for your device. • **Discoverable**: Set your device to be visible to other Bluetooth devices. • **Scan for devices**: Search for available Bluetooth devices. Mobile networks • **Data roaming**: Set the device to connect another network when you are roaming or your home network is not available. • **Use only 2G networks**: Set the device to connect only to a 2G network. • **Network operators**: Search for available networks and select a network for roaming. • **Access Point Names**: Set up access point names (APNs). Airplane mode Disable all wireless functions on your device. **JuiceDefender App** SAMSUNG_PRIORART0000379 (Latedroid):

’701 Claims	Android Device with One or More Apps
	JuiceDefender  19:19 JuiceDefender Juice Control Enabled APN MMS WiFi AutoWiFi Triggers Screen Battery Schedule Night 5m 15m 30m 1h 2h Quick Help JuicePlotter Location Disable WiFi when far from known networks Screen Enable APN+WiFi when screen on Battery Disable APN+WiFi when battery low Night Disable APN+WiFi from 22:00 to 06:00 Schedule Enable APN+WiFi 1m every 5m Activity log 7:16 PM JuiceDefender enabled: APN - battery, scheduler 2m/15m, screen 7:16 PM JuiceDefender enabled: APN - battery, scheduler 2m/15m, screen 7:16 PM Schedule start JuiceDefender saves battery power (*lots of it!*) by controlling the device data connection and/or WiFi. You can schedule regular APN/WiFi activation to let background data sync occur and have APN/WiFi enabled while the screen is on. It also helps in minimizing distractions ;) The *Easy Mode* is a no-fuss one-click way to let your battery last longer - much longer. Just enable JuiceDefender by clicking on the big button and you're ready to go! If you want more fine-grained control, try *Advanced Mode*, where you can configure all JuiceDefender features to your liking.

'701 Claims	Android Device with One or More Apps
	 There are 5 *triggers* for the enable/disable behaviour: **Battery** - when battery level gets low (less than 15%), disable APN/WiFi, and re-enable them when battery level is restored. APN/WiFi will also be enabled while the device is being recharged. **Schedule** - regularly enable APN/WiFi for a short period of time, to *let background data sync* occur (email, Twitter, Facebook, stock quotes...). If *Quick* is disabled APN/WiFi stays enabled for a longer period, useful if your data connection is very slow or you need to sync lots of data. **Night schedule** (requires *UltimateJuice*) - disable APN/WiFi during night time; you can also optionally put the phone in Silent Mode. **Screen** - enable APN/WiFi *while the screen is on* to allow browsing, tweeting, procrastination and general internet-powered enjoyment, regardless of scheduled events and battery level. **Location** (requires *UltimateJuice*) - this trigger controlled by the *'AutoWiFi'* button. It disables WiFi when the device is not in range of any known WiFi network. The location is determined via the cellular network, so it's usually quite coarse. It's a fully automatic set-it-and-forget-it WiFi manager! The *priority order* of the triggers is 1) location (WiFi only), 2) screen, 3) battery, 4) night schedule, 5) schedule - this means, for example, that when the screen is on APN/WiFi will be enabled even when the battery is low, or that the regular schedule won't occur during the night period. SAMSUNG_PRIORART0000361 (Purdy):

'701 Claims	Android Device with One or More Apps
	 Juice Control Enabled APN MMS WiFi AutoWiFi Triggers Screen Battery Schedule Night Sil 5m 15m 30m 1h 2h Quick Location Disable WiFi when far from known network: Screen Enable APN+MMS when screen on Battery Disable APN+MMS when battery low Night Disable APN+MMS from 10pm to 6:30am Schedule Enable APN+MMS 2m every 15m 9:36 AM Screen ON 9:19 AM Screen OFF 9:19 AM Night schedule end Android: Most phones don't make it easy to switch cellular data connection on and off, even if doing so really helps save your battery. JuiceDefender toggles wireless data and Wi-Fi on and off every so often to preserve power. The whole point of a smartphone with Google apps baked in is constant connectivity, of course, and you don't want to shut off access to your email, Google Voice messages, and other online services. But when you're walking around, at your office desk, and generally not actively using your phone, you probably don't need your phone to check in every minute with the mothership. JuiceDefender lets you set a time interval—5 minutes, 15, 30, an hour, two hours—at which its background process will re-enable your carrier APN, see if there are new messages or data coming in, and then shut off again. You can also set similar Wi-Fi connectivity rules, or only have web data enabled when you've got your screen on. Besides the battery savings, those who like to parse out their email checks and avoid minute-by-minute distractions see some benefit here, too. *See*, *e.g.*, SAMSUNG_PRIORART0000351 (Configuration-Translated):  Schedule Enable APN for 5m every 30m "• **Schedule:** used to configure how often Juice Defender will activate the APN / WIFI to synchronize emails, social networks, etc .., we select the interval in the bar of your submenu for example 5 minutes every half hour also has the **Quick** button to minimize the connection time for the previous example

’701 Claims	Android Device with One or More Apps
	would be 2 minutes every half hour, I recommend not using this option if the selected time period is too short, it may not give you time to connect.” *See*, *e.g.*, SAMSUNG_PRIORART0000361 (Purdy):  Schedule Enable APN+MMS 2m every 15m “JuiceDefender lets you set a time interval—5 minutes, 15, 30, an hour, two hours— at which its background process will re-enable your carrier APN, see if there are new messages or data coming in, and then shut off again.” *See*, *e.g.*, SAMSUNG_PRIORART0000335 (Ruddock):  Schedule Enable Data/WiFi for 1m every 15m **GreenPower App** POUZERATE0000196 (GreenPower User Guide) (“Global Wireless settings This is a shortcut to the phone system wireless settings where the user can find the setting “Mobile Network”. That one should be checked or Green Power won't be able to properly manage Mobile Network. Screen off wireless delay This setting defines how long Green Power should wait before switching off wireless when the screen is turned off. Delaying turning off wireless is useful for instance if the user is reading something on the screen, not touching it. At some point the screen might turns off and you will touch it or press some buttons to switch it on again. Therefore, the wireless shouldn't be interrupted here. So, instead of switching off the wireless at once when the screen turns off, Green Power will wait that this delay elapses before switching off the wireless.

'701 Claims	Android Device with One or More Apps
	Wireless on delay This setting defines how long Green Power keeps the wireless on before turning it off again. This applies to the Wifi is the setting "Manage Wifi" is selected, and this applies to the Mobile Network if the setting "Manage Mobile Network" is selected. Wireless off delay This setting defines how long Green Power keeps the wireless off before turning it on again. This applies to the Wifi is the setting "Manage Wifi" is selected, and this applies to the Mobile Network if the setting "Manage Mobile Network" is selected. Screen on setting If this is selected, the wireless will be kept on when the screen is on. This applies to the Wifi is the setting "Manage Wifi" is selected, and this applies to the Mobile Network if the setting "Manage Mobile Network" is selected. If this is not selected, then Green Power will not make any difference whether the screen is on or off:: It will regularly switch on and off wireless if needed even if the screen is on. This can be useful if the you are using the phone for anything else than using wireless data (calling, playing local game, etc). In such a case you don't need the wireless to be always on. Power on setting If this is selected, the wireless will be kept on when the phone is connected to a power source. This applies to the Wifi is the setting "Manage Wifi" is selected, and this applies to the Mobile Network if the setting "Manage Mobile Network" is selected. This overrides the "Screen on setting": If this is selected and the power is connected, then wireless will be kept on whatever the screen state is. If this is not selected, then Green Power will not make any difference whether the phone is connected to the power or not:: It will regularly switch on and off wireless if needed.

'701 Claims	Android Device with One or More Apps
	Check Traffic If this is selected, then prior to turning off wireless, Green Power will check that there is no network traffic. If there is, it will wait a few seconds and checks again until there is no traffic anymore.")
[2] The wireless end-user device of claim 1, wherein the one or more processors are configured to determine that the first end-user application is running in a background state when a user of the device is not directly interacting with that application or perceiving any benefit from that application.	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for claims [1 pre] – [1e].
[3] The wireless end-user device of claim 1, further comprising a user interface to provide a user of the device with information regarding why the first differential traffic control policy is	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for claims [1 pre] – [1e].

'701 Claims	Android Device with One or More Apps
applied to the first end-user application.	
[4] The wireless end-user device of claim 1, further comprising a user interface to inform a user of the device when there are options to set, control, override, or modify service usage controls that affect the first differential traffic control policy.	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See*, *e.g.*, the disclosures identified for claims [1 pre] – [1e].
[5] The wireless end-user device of claim 1, wherein the first differential traffic control policy is part of a multimode profile having different policies for different networks.	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See*, *e.g.*, the disclosures identified for claims [1 pre] – [1e].
[6] The wireless end-user device of claim 5, wherein the one or more processors are further configured to select a traffic control policy	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See*, *e.g.*, the disclosures identified for claims [1 pre] – [1e].

’701 Claims	Android Device with One or More Apps
from the multimode profile based at least in part on the type of network connection currently in use by the device.	
[7] The wireless end-user device of claim 6, wherein the one or more processors are further configured to, when the type of network connection is at least one type of WLAN connection, select a traffic control policy from the multimode profile based at least in part on a type of network connection from the WLAN to the Internet.	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for claims [1 pre] – [1e].
[8] The wireless end-user device of claim 1, wherein the one or more processors are further configured to determine whether a second end-	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for claims [1 pre] – [1e].

'701 Claims	Android Device with One or More Apps
user application is running in a background state or as a foreground application, and control application access for Internet service activities by the second end-user application similar to the control of application access for the first end-user application, except the control of application access for Internet service activities by the second end-user application is based on a second differential traffic control policy that can be set different from the first differential traffic control policy.	
[9] The wireless end-user device of claim 1, further comprising a network stack interface integrated with the API.	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See*, *e.g.*, the disclosures identified for claims [1 pre] – [1e]. In addition, see, *e.g.*: **Android 1.0**

'701 Claims	Android Device with One or More Apps
	SAMSUNG_PRIORART0005487, Socket.cpp

```
    #include <utils/Socket.h>
    #include <utils/inet_address.h>
    #include <utils/Log.h>
    #include <utils/Timers.h>
    #ifndef HAVE_WINSOCK
    # include <sys/types.h>
    # include <sys/socket.h>
    # include <netinet/in.h>
    # include <arpa/inet.h>
    #endif
    #include <stdlib.h>
    #include <stdio.h>
    #include <unistd.h>
    #include <string.h>
    #include <errno.h>
    #include <assert.h>
    using namespace android;
    /*
     *
==========================================================================
     *      Socket
     *
==========================================================================
     */
    #ifndef INVALID_SOCKET
    # define INVALID_SOCKET (-1)
    #endif
```

'701 Claims	Android Device with One or More Apps

```
#define UNDEF_SOCKET   ((unsigned long) INVALID_SOCKET)
/*static*/ bool Socket::mBootInitialized = false;
/*
 * Extract system-dependent error code.
 */
static inline int getSocketError(void) {
#ifdef HAVE_WINSOCK
    return WSAGetLastError();
#else
    return errno;
#endif
}
/*
 * One-time initialization for socket code.
 */
/*static*/ bool Socket::bootInit(void)
{
#ifdef HAVE_WINSOCK
    WSADATA wsaData;
    int err;
    err = WSAStartup(MAKEWORD(2, 0), &wsaData);
    if (err != 0) {
        LOG(LOG_ERROR, "socket", "Unable to start WinSock\n");
        return false;
    }
    LOG(LOG_INFO, "socket", "Using WinSock v%d.%d\n",
        LOBYTE(wsaData.wVersion), HIBYTE(wsaData.wVersion));
#endif
    mBootInitialized = true;
    return true;
```

'701 Claims	Android Device with One or More Apps

	}
	/*
	 * One-time shutdown for socket code.
	 */
	/*static*/ void Socket::finalShutdown(void)
	{
	#ifdef HAVE_WINSOCK
	    WSACleanup();
	#endif
	    mBootInitialized = false;
	}
	/*
	 * Simple constructor. Allow the application to create us and then make
	 * bind/connect calls.
	 */
	Socket::Socket(void)
	    : mSock(UNDEF_SOCKET)
	{
	    if (!mBootInitialized)
	        LOG(LOG_WARN, "socket", "WARNING: sockets not initialized\n");
	}
	/*
	 * Destructor. Closes the socket and resets our storage.
	 */
	Socket::~Socket(void)
	{
	    close();
	}
	/*
	 * Create a socket and connect to the specified host and port.

’701 Claims	Android Device with One or More Apps

	*/
	int Socket::connect(const char* host, int port)
	{
	    if (mSock != UNDEF_SOCKET) {
	        LOG(LOG_WARN, "socket", "Socket already connected\n");
	        return -1;
	    }
	    InetSocketAddress sockAddr;
	    if (!sockAddr.create(host, port))
	        return -1;
	    //return doConnect(sockAddr);
	    int foo;
	    foo = doConnect(sockAddr);
	    return foo;
	}
	/*
	 * Create a socket and connect to the specified host and port.
	 */
	int Socket::connect(const InetAddress* addr, int port)
	{
	    if (mSock != UNDEF_SOCKET) {
	        LOG(LOG_WARN, "socket", "Socket already connected\n");
	        return -1;
	    }
	    InetSocketAddress sockAddr;
	    if (!sockAddr.create(addr, port))
	        return -1;
	    return doConnect(sockAddr);
	}
	/*

’701 Claims	Android Device with One or More Apps

```
 * Finish creating a socket by connecting to the remote host.
 *
 * Returns 0 on success.
 */
int Socket::doConnect(const InetSocketAddress& sockAddr)
{
#ifdef HAVE_WINSOCK
    SOCKET sock;
#else
    int sock;
#endif
    const InetAddress* addr = sockAddr.getAddress();
    int port = sockAddr.getPort();
    struct sockaddr_in inaddr;
    DurationTimer connectTimer;
    assert(sizeof(struct sockaddr_in) == addr->getAddressLength());
    memcpy(&inaddr, addr->getAddress(), addr->getAddressLength());
    inaddr.sin_port = htons(port);
    //fprintf(stderr, "--- connecting to %s:%d\n",
    //    sockAddr.getHostName(), port);
    sock = ::socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
    if (sock == INVALID_SOCKET) {
        int err = getSocketError();
        LOG(LOG_ERROR, "socket", "Unable to create socket (err=%d)\n", err);
        return (err != 0) ? err : -1;
    }
    connectTimer.start();
    if (::connect(sock, (struct sockaddr*) &inaddr, sizeof(inaddr)) != 0) {
        int err = getSocketError();
        LOG(LOG_WARN, "socket", "Connect to %s:%d failed: %d\n",
```

'701 Claims	Android Device with One or More Apps

	sockAddr.getHostName(), port, err);
	        return (err != 0) ? err : -1;
	    }
	    connectTimer.stop();
	    if ((long) connectTimer.durationUsecs() > 100000) {
	        LOG(LOG_INFO, "socket",
	            "Connect to %s:%d took %.3fs\n", sockAddr.getHostName(),
	            port, ((long) connectTimer.durationUsecs()) / 1000000.0);
	    }
	    mSock = (unsigned long) sock;
	    LOG(LOG_VERBOSE, "socket",
	        "--- connected to %s:%d\n", sockAddr.getHostName(), port);
	    return 0;
	}
	/*
	 * Close the socket if it needs closing.
	 */
	bool Socket::close(void)
	{
	    if (mSock != UNDEF_SOCKET) {
	        //fprintf(stderr, "--- closing socket %lu\n", mSock);
	#ifdef HAVE_WINSOCK
	        if (::closesocket((SOCKET) mSock) != 0)
	            return false;
	#else
	        if (::close((int) mSock) != 0)
	            return false;
	#endif
	    }
	    mSock = UNDEF_SOCKET;

'701 Claims	Android Device with One or More Apps

	return true;
	}
	/*
	 * Read data from socket.
	 *
	 * Standard semantics: read up to "len" bytes into "buf". Returns the
	 * number of bytes read, or less than zero on error.
	 */
	int Socket::read(void* buf, ssize_t len) const
	{
	    if (mSock == UNDEF_SOCKET) {
	        LOG(LOG_ERROR, "socket", "ERROR: read on invalid socket\n");
	        return -500;
	    }
	#ifdef HAVE_WINSOCK
	    SOCKET sock = (SOCKET) mSock;
	#else
	    int sock = (int) mSock;
	#endif
	    int cc;
	    cc = recv(sock, (char*)buf, len, 0);
	    if (cc < 0) {
	        int err = getSocketError();
	        return (err > 0) ? -err : -1;
	    }
	    return cc;
	}
	/*
	 * Write data to a socket.
	 *

'701 Claims	Android Device with One or More Apps

	* Standard semantics: write up to "len" bytes into "buf". Returns the
	* number of bytes written, or less than zero on error.
	*/
	int Socket::write(const void* buf, ssize_t len) const
	{
	    if (mSock == UNDEF_SOCKET) {
	        LOG(LOG_ERROR, "socket", "ERROR: write on invalid socket\n");
	        return -500;
	    }
	#ifdef HAVE_WINSOCK
	    SOCKET sock = (SOCKET) mSock;
	#else
	    int sock = (int) mSock;
	#endif
	    int cc;
	    cc = send(sock, (const char*)buf, len, 0);
	    if (cc < 0) {
	        int err = getSocketError();
	        return (err > 0) ? -err : -1;
	    }
	    return cc;
	
	Android 1.6
	
	SAMSUNG_PRIORART0005350, Socket.cpp
	
	//
	// Internet address class.
	//

'701 Claims	Android Device with One or More Apps

	#ifdef HAVE_WINSOCK
	// This needs to come first, or Cygwin gets concerned about a potential
	// clash between WinSock and <sys/types.h>.
	# include <winsock2.h>
	#endif
	
	#include <utils/Socket.h>
	#include <utils/inet_address.h>
	#include <utils/Log.h>
	#include <utils/Timers.h>
	
	#ifndef HAVE_WINSOCK
	# include <sys/types.h>
	# include <sys/socket.h>
	# include <netinet/in.h>
	# include <arpa/inet.h>
	#endif
	
	#include <stdlib.h>
	#include <stdio.h>
	#include <unistd.h>
	#include <string.h>
	#include <errno.h>
	#include <assert.h>
	
	using namespace android;
	
	/*

'701 Claims	Android Device with One or More Apps

	*
	==
	* Socket
	*
	==
	*/
	
	#ifndef INVALID_SOCKET
	# define INVALID_SOCKET (-1)
	#endif
	#define UNDEF_SOCKET ((unsigned long) INVALID_SOCKET)
	
	/*static*/ bool Socket::mBootInitialized = false;
	
	/*
	 * Extract system-dependent error code.
	 */
	static inline int getSocketError(void) {
	#ifdef HAVE_WINSOCK
	    return WSAGetLastError();
	#else
	    return errno;
	#endif
	}
	
	/*
	 * One-time initialization for socket code.
	 */
	/*static*/ bool Socket::bootInit(void)
	{

<table>
<tr><th>'701 Claims</th><th>Android Device with One or More Apps</th></tr>
<tr><td></td><td><pre>
#ifdef HAVE_WINSOCK
    WSADATA wsaData;
    int err;

    err = WSAStartup(MAKEWORD(2, 0), &wsaData);
    if (err != 0) {
        LOG(LOG_ERROR, "socket", "Unable to start WinSock\n");
        return false;
    }

    LOG(LOG_INFO, "socket", "Using WinSock v%d.%d\n",
        LOBYTE(wsaData.wVersion), HIBYTE(wsaData.wVersion));
#endif

    mBootInitialized = true;
    return true;
}

/*
 * One-time shutdown for socket code.
 */
/*static*/ void Socket::finalShutdown(void)
{
#ifdef HAVE_WINSOCK
    WSACleanup();
#endif
    mBootInitialized = false;
}
</pre></td></tr>
</table>

’701 Claims	Android Device with One or More Apps

	/*
	 * Simple constructor. Allow the application to create us and then make
	 * bind/connect calls.
	 */
	Socket::Socket(void)
	    : mSock(UNDEF_SOCKET)
	{
	    if (!mBootInitialized)
	        LOG(LOG_WARN, "socket", "WARNING: sockets not initialized\n");
	}
	
	/*
	 * Destructor. Closes the socket and resets our storage.
	 */
	Socket::~Socket(void)
	{
	    close();
	}
	
	/*
	 * Create a socket and connect to the specified host and port.
	 */
	int Socket::connect(const char* host, int port)
	{
	    if (mSock != UNDEF_SOCKET) {
	        LOG(LOG_WARN, "socket", "Socket already connected\n");
	        return -1;
	    }

'701 Claims	Android Device with One or More Apps

	InetSocketAddress sockAddr;
	    if (!sockAddr.create(host, port))
	        return -1;
	
	    //return doConnect(sockAddr);
	    int foo;
	    foo = doConnect(sockAddr);
	    return foo;
	}
	
	/*
	 * Create a socket and connect to the specified host and port.
	 */
	int Socket::connect(const InetAddress* addr, int port)
	{
	    if (mSock != UNDEF_SOCKET) {
	        LOG(LOG_WARN, "socket", "Socket already connected\n");
	        return -1;
	    }
	
	    InetSocketAddress sockAddr;
	    if (!sockAddr.create(addr, port))
	        return -1;
	
	    return doConnect(sockAddr);
	}
	
	/*
	 * Finish creating a socket by connecting to the remote host.
	 *

<table>
<tr><th>'701 Claims</th><th>Android Device with One or More Apps</th></tr>
<tr><td></td><td><pre>
 * Returns 0 on success.
 */
int Socket::doConnect(const InetSocketAddress& sockAddr)
{
#ifdef HAVE_WINSOCK
    SOCKET sock;
#else
    int sock;
#endif
    const InetAddress* addr = sockAddr.getAddress();
    int port = sockAddr.getPort();
    struct sockaddr_in inaddr;
    DurationTimer connectTimer;

    assert(sizeof(struct sockaddr_in) == addr->getAddressLength());
    memcpy(&inaddr, addr->getAddress(), addr->getAddressLength());
    inaddr.sin_port = htons(port);

    //fprintf(stderr, "--- connecting to %s:%d\n",
    //   sockAddr.getHostName(), port);

    sock = ::socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
    if (sock == INVALID_SOCKET) {
        int err = getSocketError();
        LOG(LOG_ERROR, "socket", "Unable to create socket (err=%d)\n", err);
        return (err != 0) ? err : -1;
    }

    connectTimer.start();
</pre></td></tr>
</table>

’701 Claims	Android Device with One or More Apps

	if (::connect(sock, (struct sockaddr*) &inaddr, sizeof(inaddr)) != 0) {
	        int err = getSocketError();
	        LOG(LOG_WARN, "socket", "Connect to %s:%d failed: %d\n",
	            sockAddr.getHostName(), port, err);
	        return (err != 0) ? err : -1;
	    }
	
	    connectTimer.stop();
	    if ((long) connectTimer.durationUsecs() > 100000) {
	        LOG(LOG_INFO, "socket",
	            "Connect to %s:%d took %.3fs\n", sockAddr.getHostName(),
	            port, ((long) connectTimer.durationUsecs()) / 1000000.0);
	    }
	
	    mSock = (unsigned long) sock;
	    LOG(LOG_VERBOSE, "socket",
	        "--- connected to %s:%d\n", sockAddr.getHostName(), port);
	    return 0;
	}
	
	/*
	 * Close the socket if it needs closing.
	 */
	bool Socket::close(void)
	{
	    if (mSock != UNDEF_SOCKET) {
	        //fprintf(stderr, "--- closing socket %lu\n", mSock);
	#ifdef HAVE_WINSOCK
	        if (::closesocket((SOCKET) mSock) != 0)

'701 Claims	Android Device with One or More Apps

	return false;
	#else
	    if (::close((int) mSock) != 0)
	        return false;
	#endif
	    }
	
	    mSock = UNDEF_SOCKET;
	
	    return true;
	}
	
	/*
	 * Read data from socket.
	 *
	 * Standard semantics: read up to "len" bytes into "buf". Returns the
	 * number of bytes read, or less than zero on error.
	 */
	int Socket::read(void* buf, ssize_t len) const
	{
	    if (mSock == UNDEF_SOCKET) {
	        LOG(LOG_ERROR, "socket", "ERROR: read on invalid socket\n");
	        return -500;
	    }
	
	#ifdef HAVE_WINSOCK
	    SOCKET sock = (SOCKET) mSock;
	#else
	    int sock = (int) mSock;
	#endif

’701 Claims	Android Device with One or More Apps

	int cc;
	
	    cc = recv(sock, (char*)buf, len, 0);
	    if (cc < 0) {
	        int err = getSocketError();
	        return (err > 0) ? -err : -1;
	    }
	
	    return cc;
	}
	
	/*
	 * Write data to a socket.
	 *
	 * Standard semantics: write up to "len" bytes into "buf". Returns the
	 * number of bytes written, or less than zero on error.
	 */
	int Socket::write(const void* buf, ssize_t len) const
	{
	    if (mSock == UNDEF_SOCKET) {
	        LOG(LOG_ERROR, "socket", "ERROR: write on invalid socket\n");
	        return -500;
	    }
	
	#ifdef HAVE_WINSOCK
	    SOCKET sock = (SOCKET) mSock;
	#else
	    int sock = (int) mSock;
	#endif
	    int cc;

'701 Claims	Android Device with One or More Apps

	cc = send(sock, (const char*)buf, len, 0);
	if (cc < 0) {
	int err = getSocketError();
	return (err > 0) ? -err : -1;
	}
	
	return cc;
	}
	
	Android 2.2
	
	SAMSUNG_PRIORART00050605~~5353~~, Socket
	
	/**
	* Creates a new unconnected socket. When a SocketImplFactory is defined it
	* creates the internal socket implementation, otherwise the default socket
	* implementation will be used for this socket.
	*
	* @see SocketImplFactory
	* @see SocketImpl
	*/
	
	/**
	* Tries to connect a socket to all IP addresses of the given hostname.
	*
	* @param dstName
	* the target host name or IP address to connect to.
	* @param dstPort

'701 Claims	Android Device with One or More Apps
	* the port on the target host to connect to. * @param localAddress * the address on the local host to bind to. * @param localPort * the port on the local host to bind to. * @param streaming * if {@code true} a streaming socket is returned, a datagram * socket otherwise. * @throws UnknownHostException * if the host name could not be resolved into an IP address. * @throws IOException * if an error occurs while creating the socket. * @throws SecurityException * if a security manager exists and it denies the permission to * connect to the given address and port. */ /** * Creates a new streaming socket connected to the target host specified by * the parameters {@code dstName} and {@code dstPort}. The socket is bound * to any available port on the local host. * <p><strong>Implementation note:</strong> this implementation tries each * IP address for the given hostname until it either connects successfully * or it exhausts the set. It will try both IPv4 and IPv6 addresses in the * order specified by the system property {@code "java.net.preferIPv6Addresses"}. * * @param dstName * the target host name or IP address to connect to. * @param dstPort * the port on the target host to connect to.

'701 Claims	Android Device with One or More Apps
	* @throws UnknownHostException * if the host name could not be resolved into an IP address. * @throws IOException * if an error occurs while creating the socket. * @throws SecurityException * if a security manager exists and it denies the permission to * connect to the given address and port. */ /** * Checks whether the connection destination satisfies the security policy * and the validity of the port range. * * @param destAddr * the destination host address. * @param dstPort * the port on the destination host. */ /** * Closes the socket. It is not possible to reconnect or rebind to this * socket thereafter which means a new socket instance has to be created. * * @throws IOException * if an error occurs while closing the socket. */ /** * Gets the IP address of the target host this socket is connected to. *

'701 Claims	Android Device with One or More Apps
	* @return the IP address of the connected target host or {@code null} if * this socket is not yet connected. */ /** * Gets the local IP address this socket is bound to. * * @return the local IP address of this socket or {@code InetAddress.ANY} if * the socket is unbound. */ /** * Gets the local port this socket is bound to. * * @return the local port of this socket or {@code -1} if the socket is * unbound. */ /** * Gets the port number of the target host this socket is connected to. * * @return the port number of the connected target host or {@code 0} if this * socket is not yet connected. */ /** * Gets the local address and port of this socket as a SocketAddress or * {@code null} if the socket is unbound. This is useful on multihomed * hosts. *

'701 Claims	Android Device with One or More Apps
	* @return the bound local socket address and port. */ /** * Binds this socket to the given local host address and port specified by * the SocketAddress {@code localAddr}. If {@code localAddr} is set to * {@code null}, this socket will be bound to an available local address on * any free port. * * @param localAddr * the specific address and port on the local machine to bind to. * @throws IllegalArgumentException * if the given SocketAddress is invalid or not supported. * @throws IOException * if the socket is already bound or an error occurs while * binding. */ /** * Connects this socket to the given remote host address and port specified * by the SocketAddress {@code remoteAddr}. * * @param remoteAddr * the address and port of the remote host to connect to. * @throws IllegalArgumentException * if the given SocketAddress is invalid or not supported. * @throws IOException * if the socket is already connected or an error occurs while

'701 Claims	**Android Device with One or More Apps**
	* connecting. */
[10] The wireless end-user device of claim 1, further comprising a networking stack, wherein the one or more processors are further configured to, at an application service interface layer, identify application traffic flows prior to the flows entering the networking stack.	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for claims [1 pre] – [1e], [9].
[11] The wireless end-user device of claim 1, wherein the one or more processors control of application access based on a first differential traffic control policy is applied to one of but not both of a connection to a roaming WWAN network and a	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for claims [1 pre] – [1e].

'701 Claims	Android Device with One or More Apps
connection to a home WWAN network.	
[12] The wireless end-user device of claim 1, wherein the one or more processors are further configured to dynamically change the application of the first differential traffic control policy based on a power state of the device.	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for claims [1 pre] – [1e]. In addition, see, *e.g.*: **Android 1.0** **SAMSUNG_PRIORART0005487, Power** /** * Wake lock that ensures that the CPU is running. The screen might * not be on. */ public static final int PARTIAL_WAKE_LOCK = 1; /** * Wake lock that ensures that the screen is on. */ public static final int FULL_WAKE_LOCK = 2; * Brightness value to use when battery is low */ public static final int BRIGHTNESS_LOW_BATTERY = 10; /** * Threshold for BRIGHTNESS_LOW_BATTERY (percentage) * Screen will stay dim if battery level is <= LOW_BATTERY_THRESHOLD */

'701 Claims	Android Device with One or More Apps

	public static final int LOW_BATTERY_THRESHOLD = 10;
	SAMSUNG_PRIORART0005487, BatteryManager
	// values for "status" field in the ACTION_BATTERY_CHANGED Intent
	public static final int BATTERY_STATUS_UNKNOWN = 1;
	public static final int BATTERY_STATUS_CHARGING = 2;
	public static final int BATTERY_STATUS_DISCHARGING = 3;
	public static final int BATTERY_STATUS_NOT_CHARGING = 4;
	public static final int BATTERY_STATUS_FULL = 5;
	// values for "health" field in the ACTION_BATTERY_CHANGED Intent
	public static final int BATTERY_HEALTH_UNKNOWN = 1;
	public static final int BATTERY_HEALTH_GOOD = 2;
	public static final int BATTERY_HEALTH_OVERHEAT = 3;
	public static final int BATTERY_HEALTH_DEAD = 4;
	public static final int BATTERY_HEALTH_OVER_VOLTAGE = 5;
	public static final int BATTERY_HEALTH_UNSPECIFIED_FAILURE = 6;
	// values of the "plugged" field in the ACTION_BATTERY_CHANGED intent
	public static final int BATTERY_PLUGGED_AC = 1;
	public static final int BATTERY_PLUGGED_USB = 2;
	Android 1.6
	SAMSUNG_PRIORART0005350, Power

'701 Claims	Android Device with One or More Apps
	/** * Brightness value to use when battery is low */ public static final int BRIGHTNESS_LOW_BATTERY = 10; /** * Threshold for BRIGHTNESS_LOW_BATTERY (percentage) * Screen will stay dim if battery level is <= LOW_BATTERY_THRESHOLD */ public static final int LOW_BATTERY_THRESHOLD = 10; **SAMSUNG_PRIORART0005350, PowerManager** /** * This class gives you control of the power state of the device. * * <p><b>Device battery life will be significantly affected by the use of this API.</b> Do not * acquire WakeLocks unless you really need them, use the minimum levels possible, and be sure * to release it as soon as you can. * * <p>You can obtain an instance of this class by calling * {@link android.content.Context#getSystemService(java.lang.String) Context.getSystemService()}. * * <p>The primary API you'll use is {@link #newWakeLock(int, String) newWakeLock()}. This will * create a {@link PowerManager.WakeLock} object. You can then use methods on this object to * control the power state of the device. In practice it's quite simple: * * {@samplecode * PowerManager pm = (PowerManager) getSystemService(Context.POWER_SERVICE); * PowerManager.WakeLock wl = pm.newWakeLock(PowerManager.SCREEN_DIM_WAKE_LOCK, "My Tag");

'701 Claims	Android Device with One or More Apps
	* wl.acquire(); * ..screen will stay on during this section.. * wl.release(); * } * * <p>The following flags are defined, with varying effects on system power. <i>These flags are * mutually exclusive - you may only specify one of them.</i> * <table border="2" width="85%" align="center" frame="hsides" rules="rows"> * * <thead> * <tr><th>Flag Value</th> * <th>CPU</th> <th>Screen</th> <th>Keyboard</th></tr> * </thead> * * <tbody> * <tr><th>{@link #PARTIAL_WAKE_LOCK}</th> * <td>On*</td> <td>Off</td> <td>Off</td> * </tr> * * <tr><th>{@link #SCREEN_DIM_WAKE_LOCK}</th> * <td>On</td> <td>Dim</td> <td>Off</td> * </tr> * * <tr><th>{@link #SCREEN_BRIGHT_WAKE_LOCK}</th> * <td>On</td> <td>Bright</td> <td>Off</td> * </tr> * * <tr><th>{@link #FULL_WAKE_LOCK}</th> * <td>On</td> <td>Bright</td> <td>Bright</td> * </tr> * </tbody>

'701 Claims	Android Device with One or More Apps
	* </table> * * <p>*<i>If you hold a partial wakelock, the CPU will continue to run, irrespective of any timers * and even after the user presses the power button. In all other wakelocks, the CPU will run, but * the user can still put the device to sleep using the power button.</i> * * <p>In addition, you can add two more flags, which affect behavior of the screen only. <i>These * flags have no effect when combined with a {@link #PARTIAL_WAKE_LOCK}.</i> * <table border="2" width="85%" align="center" frame="hsides" rules="rows"> * * <thead> * <tr><th>Flag Value</th> <th>Description</th></tr> * </thead> * * <tbody> * <tr><th>{@link #ACQUIRE_CAUSES_WAKEUP}</th> * <td>Normal wake locks don't actually turn on the illumination. Instead, they cause * the illumination to remain on once it turns on (e.g. from user activity). This flag * will force the screen and/or keyboard to turn on immediately, when the WakeLock is * acquired. A typical use would be for notifications which are important for the user to * see immediately.</td> * </tr> * * <tr><th>{@link #ON_AFTER_RELEASE}</th> * <td>If this flag is set, the user activity timer will be reset when the WakeLock is * released, causing the illumination to remain on a bit longer. This can be used to * reduce flicker if you are cycling between wake lock conditions.</td> * </tr> * </tbody> * </table> *

'701 Claims	Android Device with One or More Apps

	*
	 */
	
	 private static final int WAKE_BIT_CPU_STRONG = 1;
	 private static final int WAKE_BIT_CPU_WEAK = 2;
	 private static final int WAKE_BIT_SCREEN_DIM = 4;
	 private static final int WAKE_BIT_SCREEN_BRIGHT = 8;
	 private static final int WAKE_BIT_KEYBOARD_BRIGHT = 16;
	
	 SAMSUNG_PRIORART0005350, BatteryManager
	
	 /**
	 * The BatteryManager class contains strings and constants used for values
	 * in the ACTION_BATTERY_CHANGED Intent.
	 */
	 public class BatteryManager {
	
	 // values for "status" field in the ACTION_BATTERY_CHANGED Intent
	 public static final int BATTERY_STATUS_UNKNOWN = 1;
	 public static final int BATTERY_STATUS_CHARGING = 2;
	 public static final int BATTERY_STATUS_DISCHARGING = 3;
	 public static final int BATTERY_STATUS_NOT_CHARGING = 4;
	 public static final int BATTERY_STATUS_FULL = 5;
	
	 // values for "health" field in the ACTION_BATTERY_CHANGED Intent
	 public static final int BATTERY_HEALTH_UNKNOWN = 1;
	 public static final int BATTERY_HEALTH_GOOD = 2;
	 public static final int BATTERY_HEALTH_OVERHEAT = 3;
	 public static final int BATTERY_HEALTH_DEAD = 4;
	 public static final int BATTERY_HEALTH_OVER_VOLTAGE = 5;

'701 Claims	Android Device with One or More Apps

	public static final int BATTERY_HEALTH_UNSPECIFIED_FAILURE = 6;
	
	// values of the "plugged" field in the ACTION_BATTERY_CHANGED intent.
	// These must be powers of 2.
	/** Power source is an AC charger. */
	public static final int BATTERY_PLUGGED_AC = 1;
	/** Power source is a USB port. */
	public static final int BATTERY_PLUGGED_USB = 2;
	}
	
	SAMSUNG_PRIORART0005350, BatteryStats
	
	/**
	* A constant indicating a a wifi turn on timer
	*
	* {@hide}
	*/
	public static final int WIFI_TURNED_ON = 4;
	
	/**
	* A constant indicating a full wifi lock timer
	*
	* {@hide}
	*/
	public static final int FULL_WIFI_LOCK = 5;
	
	/**
	* A constant indicating a scan wifi lock timer
	*
	* {@hide}

’701 Claims	Android Device with One or More Apps
	*/ public static final int SCAN_WIFI_LOCK = 6; /** * A constant indicating a wifi multicast timer * * {@hide} */ public static final int WIFI_MULTICAST_ENABLED = 7; /** * A constant indicating an audio turn on timer * * {@hide} */ public static final int AUDIO_TURNED_ON = 7; /** * A constant indicating a video turn on timer * * {@hide} */ public static final int VIDEO_TURNED_ON = 8; /** * Include all of the data in the stats, including previously saved data. */ public static final int STATS_TOTAL = 0; /** * Include only the last run in the stats.

'701 Claims	Android Device with One or More Apps
	*/ public static final int STATS_LAST = 1; /** * Include only the current run in the stats. */ public static final int STATS_CURRENT = 2; /** * Include only the run since the last time the device was unplugged in the stats. */ public static final int STATS_UNPLUGGED = 3; public abstract void noteWifiTurnedOnLocked(); public abstract void noteWifiTurnedOffLocked(); public abstract void noteFullWifiLockAcquiredLocked(); public abstract void noteFullWifiLockReleasedLocked(); public abstract void noteScanWifiLockAcquiredLocked(); public abstract void noteScanWifiLockReleasedLocked(); public abstract void noteWifiMulticastEnabledLocked(); public abstract void noteWifiMulticastDisabledLocked(); /** * Returns the time in microseconds that the screen has been on while the device was * running on battery. * * {@hide} */ public abstract long getScreenOnTime(long batteryRealtime, int which); public static final int SCREEN_BRIGHTNESS_DARK = 0;

'701 Claims	Android Device with One or More Apps
	public static final int SCREEN_BRIGHTNESS_DIM = 1; public static final int SCREEN_BRIGHTNESS_MEDIUM = 2; public static final int SCREEN_BRIGHTNESS_LIGHT = 3; public static final int SCREEN_BRIGHTNESS_BRIGHT = 4; public static final int DATA_CONNECTION_NONE = 0; public static final int DATA_CONNECTION_GPRS = 1; public static final int DATA_CONNECTION_EDGE = 2; public static final int DATA_CONNECTION_UMTS = 3; public static final int DATA_CONNECTION_OTHER = 4; /** * Returns the time in microseconds that wifi has been on while the device was * running on battery. * * {@hide} */ public abstract long getWifiOnTime(long batteryRealtime, int which); /** * Returns the time in microseconds that bluetooth has been on while the device was * running on battery. * * {@hide} */ public abstract long getBluetoothOnTime(long batteryRealtime, int which); /** * Return whether we are currently running on battery. */ public abstract boolean getIsOnBattery();

'701 Claims	Android Device with One or More Apps
	/** * Returns the time that the radio was on for data transfers. * @return the uptime in microseconds while unplugged */ public abstract long getRadioDataUptime(); /** * Returns the current battery realtime in microseconds. * * @param curTime the amount of elapsed realtime in microseconds. */ public abstract long getBatteryRealtime(long curTime); /** * Returns the battery percentage level at the last time the device was unplugged from power, or * the last time it booted on battery power. */ public abstract int getDischargeStartLevel(); **Android 2.2** **SAMSUNG_PRIORART0005353, Power** /** * Brightness value to use when battery is low */ public static final int BRIGHTNESS_LOW_BATTERY = 10; /**

'701 Claims	Android Device with One or More Apps

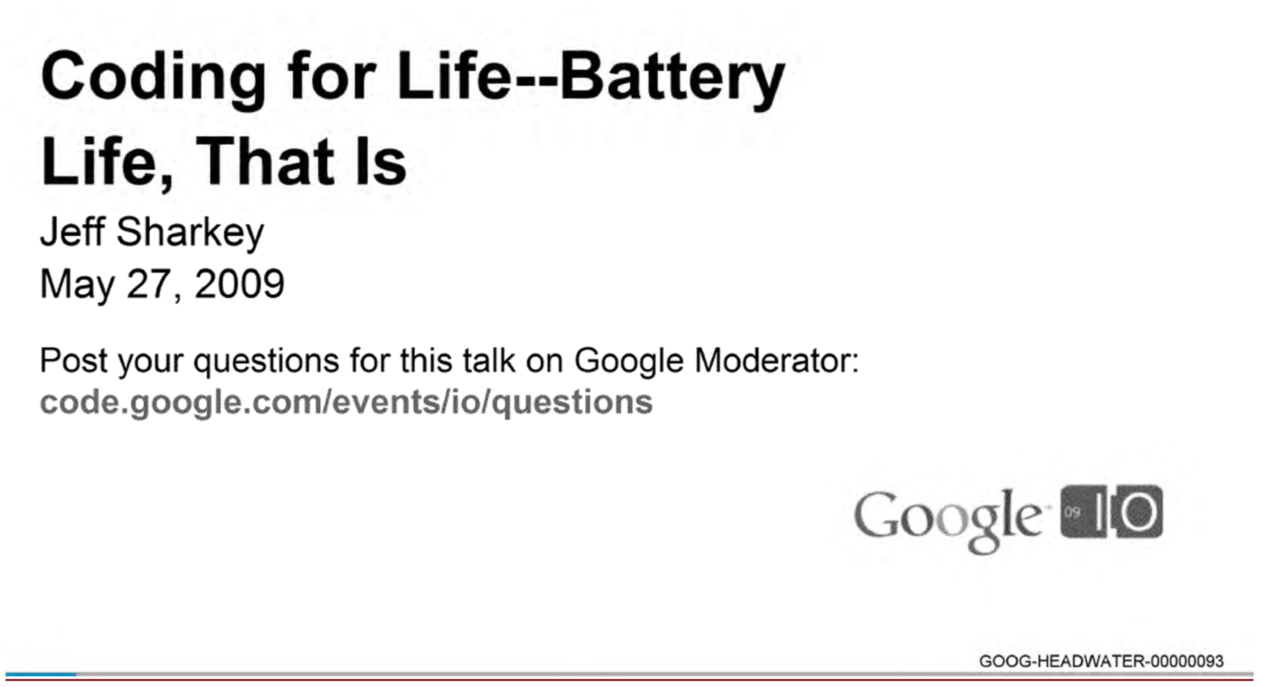
	* Threshold for BRIGHTNESS_LOW_BATTERY (percentage)
	    * Screen will stay dim if battery level is <= LOW_BATTERY_THRESHOLD
	    */
	   public static final int LOW_BATTERY_THRESHOLD = 10;
	
	   /**
	    * Turn the screen on or off
	    *
	    * @param on Whether you want the screen on or off
	    */
	   public static native int setScreenState(boolean on);
	
	SAMSUNG_PRIORART0005353, BatteryManager
	   /**
	    * Extra for {@link android.content.Intent#ACTION_BATTERY_CHANGED}:
	    * integer indicating whether the device is plugged in to a power
	    * source; 0 means it is on battery, other constants are different
	    * types of power sources.
	    */
	   public static final String EXTRA_PLUGGED = "plugged";
	
	   /**
	    * Extra for {@link android.content.Intent#ACTION_BATTERY_CHANGED}:
	    * integer containing the current battery voltage level.
	    */
	   public static final String EXTRA_VOLTAGE = "voltage";
	
	   /**
	    * Extra for {@link android.content.Intent#ACTION_BATTERY_CHANGED}:
	    * integer containing the current battery temperature.

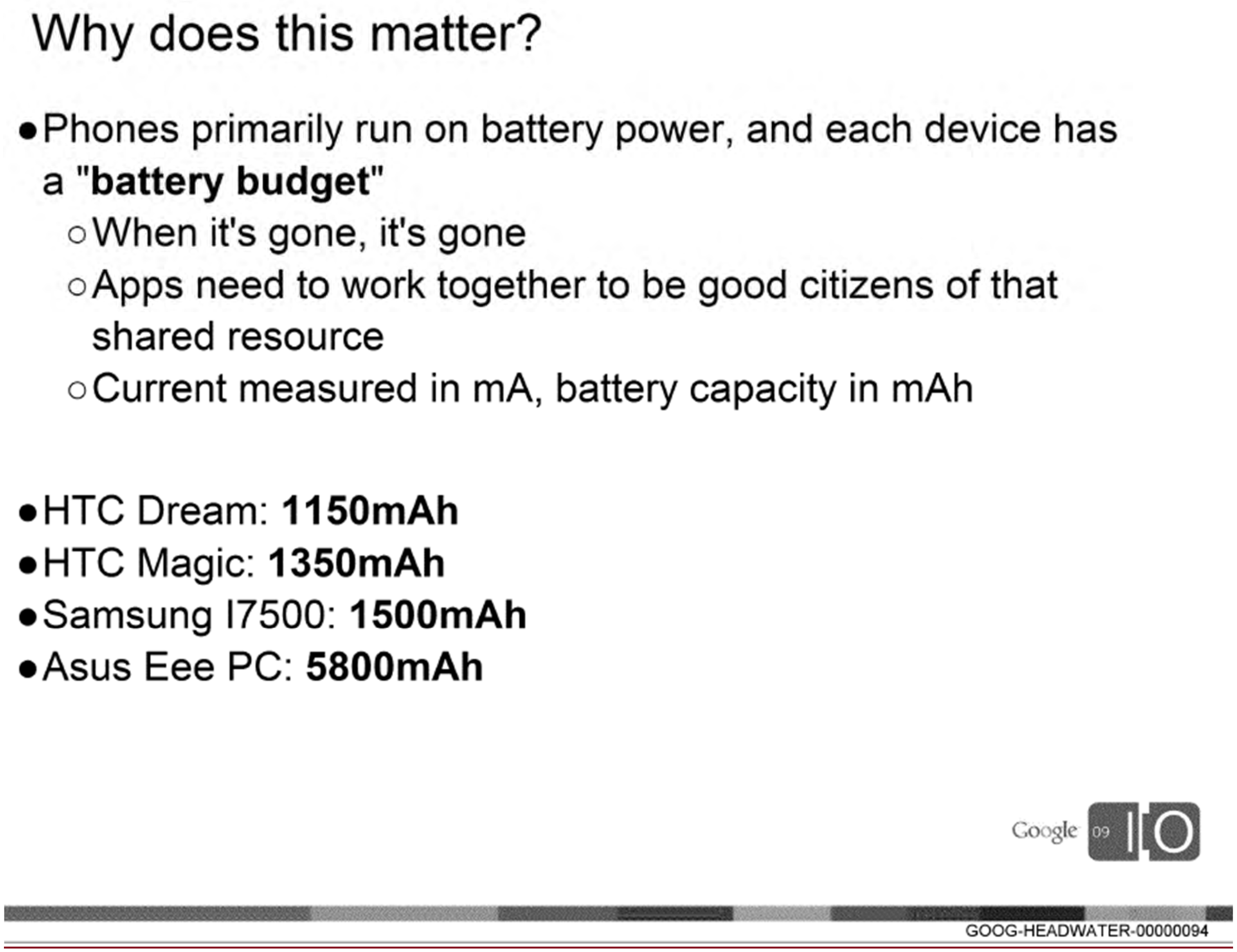
'701 Claims	Android Device with One or More Apps

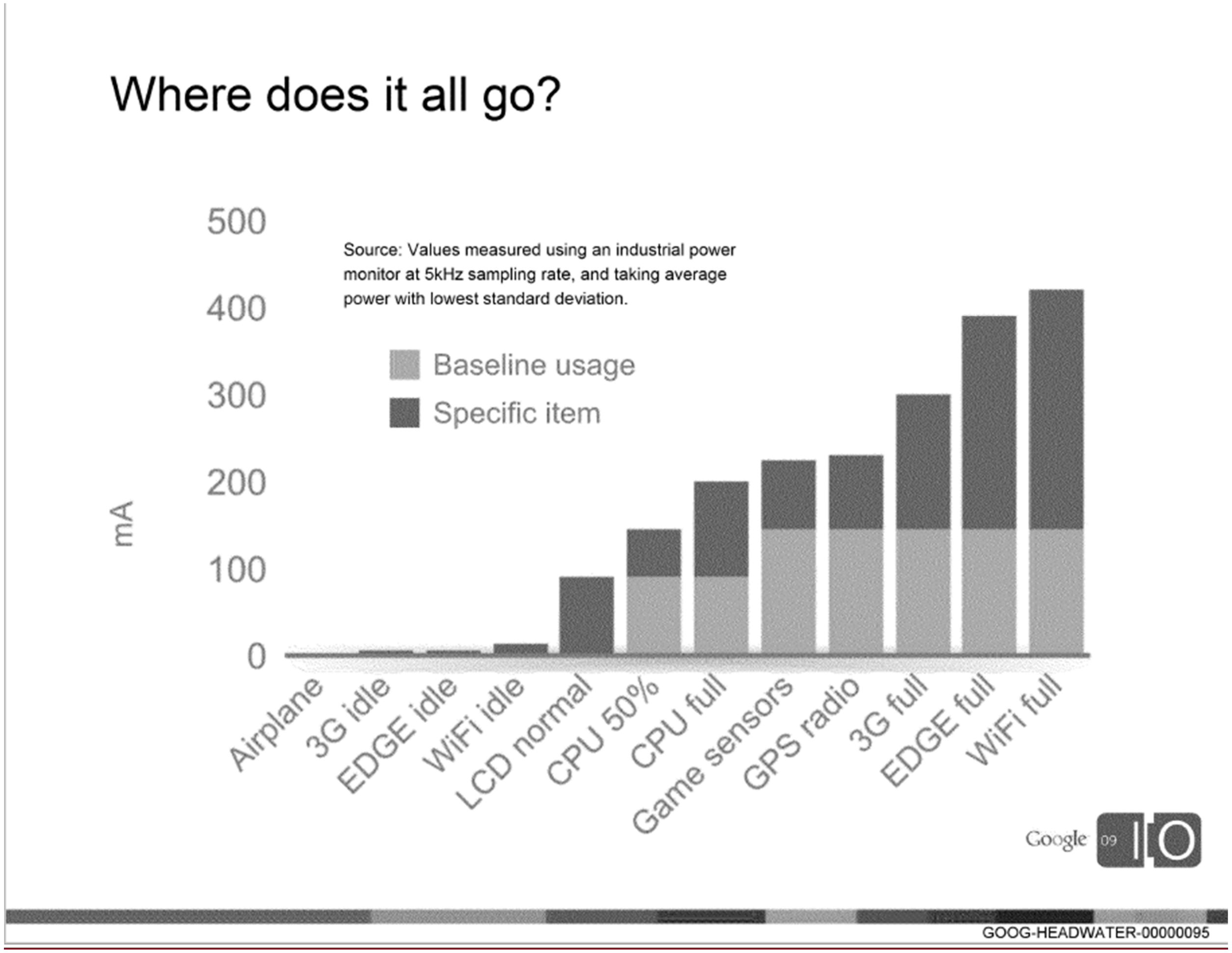
	*/
	public static final String EXTRA_TEMPERATURE = "temperature";
	
	/**
	* Extra for {@link android.content.Intent#ACTION_BATTERY_CHANGED}:
	* String describing the technology of the current battery.
	*/
	public static final String EXTRA_TECHNOLOGY = "technology";
	
	// values for "status" field in the ACTION_BATTERY_CHANGED Intent
	public static final int BATTERY_STATUS_UNKNOWN = 1;
	public static final int BATTERY_STATUS_CHARGING = 2;
	public static final int BATTERY_STATUS_DISCHARGING = 3;
	public static final int BATTERY_STATUS_NOT_CHARGING = 4;
	public static final int BATTERY_STATUS_FULL = 5;
	
	// values of the "plugged" field in the ACTION_BATTERY_CHANGED intent.
	// These must be powers of 2.
	/** Power source is an AC charger. */
	public static final int BATTERY_PLUGGED_AC = 1;
	/** Power source is a USB port. */
	public static final int BATTERY_PLUGGED_USB = 2;
	
	SAMSUNG_PRIORART0005353, BatteryStats
	
	/**
	* A class providing access to battery usage statistics, including information on
	* wakelocks, processes, packages, and services. All times are represented in microseconds
	* except where indicated otherwise.
	* @hide

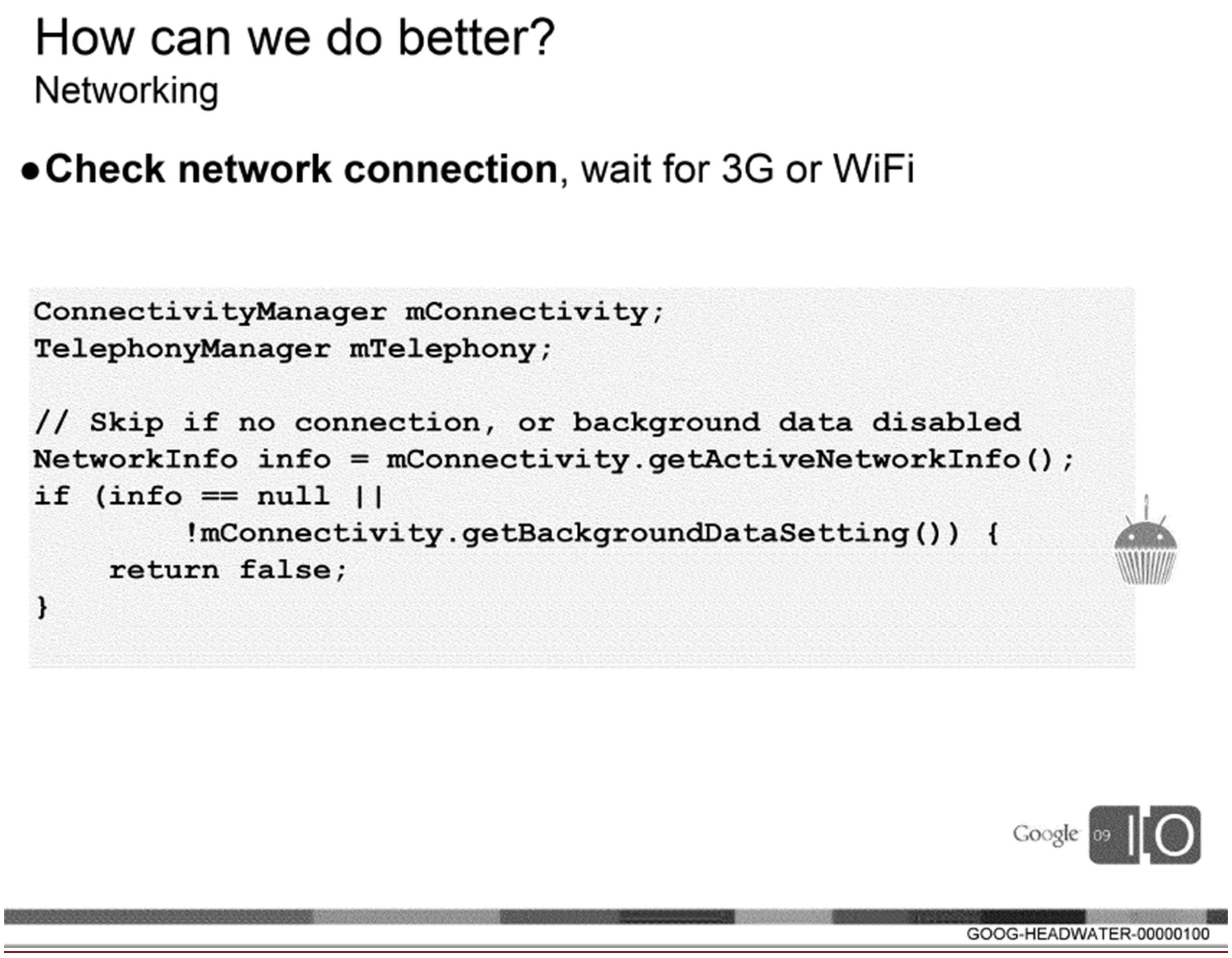
'701 Claims	Android Device with One or More Apps
	*/ /** * A constant indicating a a wifi turn on timer * * {@hide} */ public static final int WIFI_TURNED_ON = 4; /** * A constant indicating an audio turn on timer * * {@hide} */ public static final int AUDIO_TURNED_ON = 7; /** * A constant indicating a video turn on timer * * {@hide} */ public static final int VIDEO_TURNED_ON = 8; public static final int SIGNAL_STRENGTH_NONE_OR_UNKNOWN = 0; public static final int SIGNAL_STRENGTH_POOR = 1; public static final int SIGNAL_STRENGTH_MODERATE = 2; public static final int SIGNAL_STRENGTH_GOOD = 3; public static final int SIGNAL_STRENGTH_GREAT = 4; static final String[] SIGNAL_STRENGTH_NAMES = { "none", "poor", "moderate", "good", "great"

'701 Claims	Android Device with One or More Apps
	}; public static final int DATA_CONNECTION_NONE = 0; public static final int DATA_CONNECTION_GPRS = 1; public static final int DATA_CONNECTION_EDGE = 2; public static final int DATA_CONNECTION_UMTS = 3; public static final int DATA_CONNECTION_OTHER = 4; **GOOG-HEADWATER-00000092, Google I/O 2009 - Coding for Life -- Battery Life, That Is (June 2, 2009)**

’701 Claims	Android Device with One or More Apps
	See, e.g., GOOG-HEADWATER-00000092 at 2:  **Coding for Life--Battery Life, That Is** Jeff Sharkey May 27, 2009 Post your questions for this talk on Google Moderator: **code.google.com/events/io/questions** Google 09 IO GOOG-HEADWATER-00000093

'701 Claims	Android Device with One or More Apps
	See, e.g., GOOG-HEADWATER-00000092 at 3:  Why does this matter? ● Phones primarily run on battery power, and each device has a **"battery budget"** ○ When it's gone, it's gone ○ Apps need to work together to be good citizens of that shared resource ○ Current measured in mA, battery capacity in mAh ● HTC Dream: **1150mAh** ● HTC Magic: **1350mAh** ● Samsung I7500: **1500mAh** ● Asus Eee PC: **5800mAh** Google 09 IO GOOG-HEADWATER-00000094

'701 Claims	Android Device with One or More Apps
	See, e.g., GOOG-HEADWATER-00000092 at 4:  Where does it all go? Source: Values measured using an industrial power monitor at 5kHz sampling rate, and taking average power with lowest standard deviation. GOOG-HEADWATER-00000095

'701 Claims	Android Device with One or More Apps
	See, e.g., GOOG-HEADWATER-00000092 at 9: 

How can we do better?

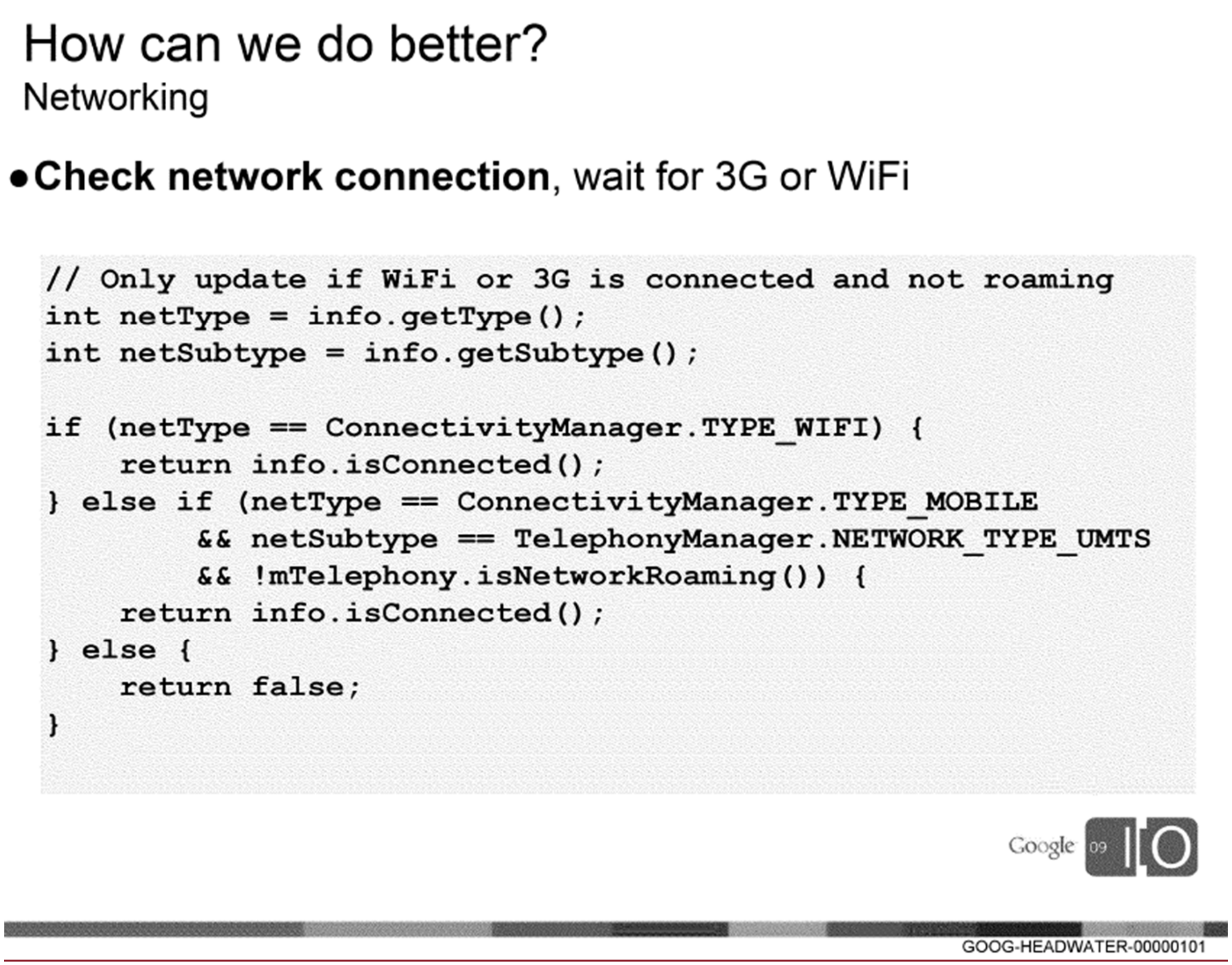
Networking

- **Check network connection**, wait for 3G or WiFi

```
ConnectivityManager mConnectivity;
TelephonyManager mTelephony;

// Skip if no connection, or background data disabled
NetworkInfo info = mConnectivity.getActiveNetworkInfo();
if (info == null ||
        !mConnectivity.getBackgroundDataSetting()) {
    return false;
}
```

Google IO 09

GOOG-HEADWATER-00000100

'701 Claims	Android Device with One or More Apps
	See, e.g., GOOG-HEADWATER-00000092 at 11: 

How can we do better?

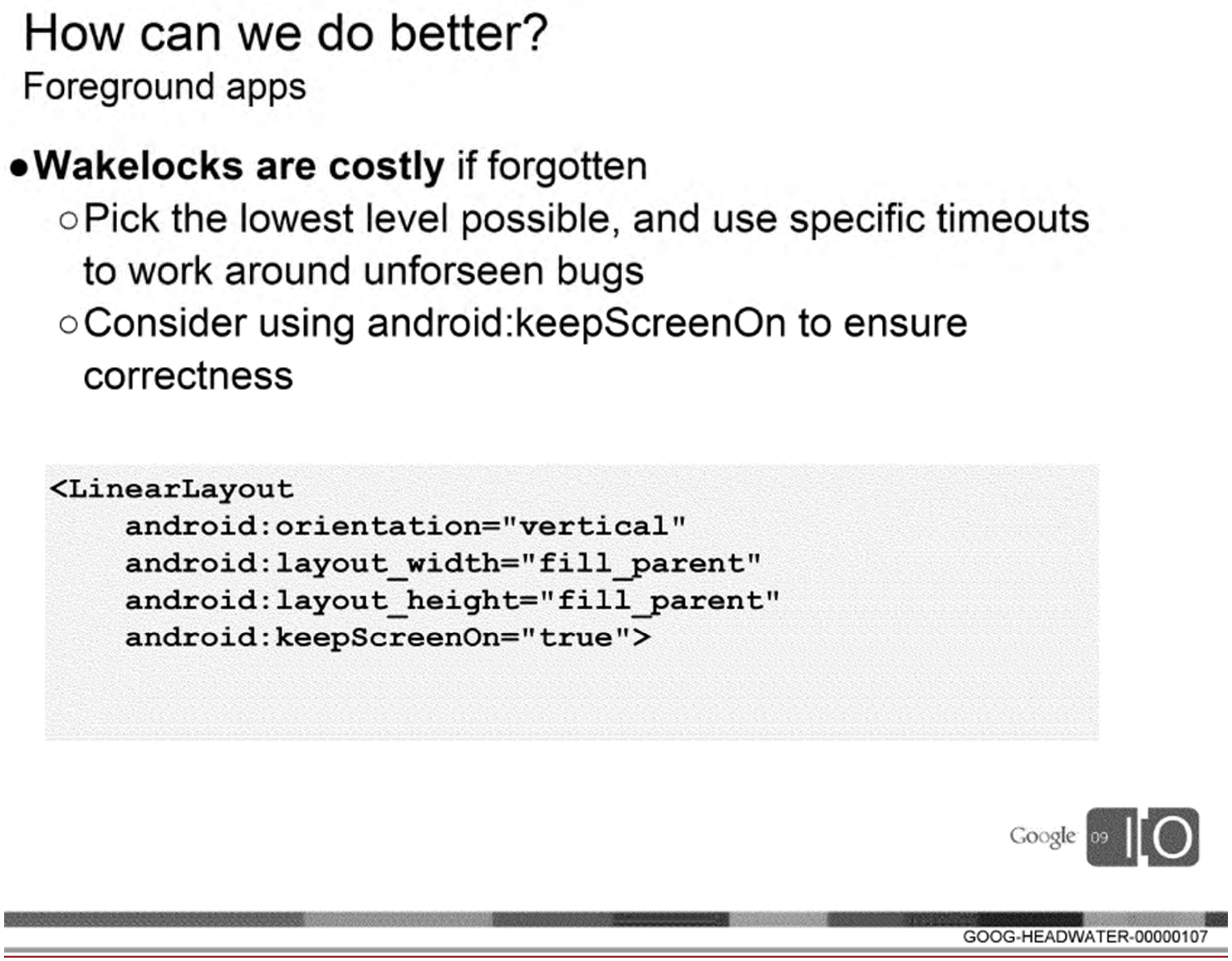
Networking

- **Check network connection**, wait for 3G or WiFi

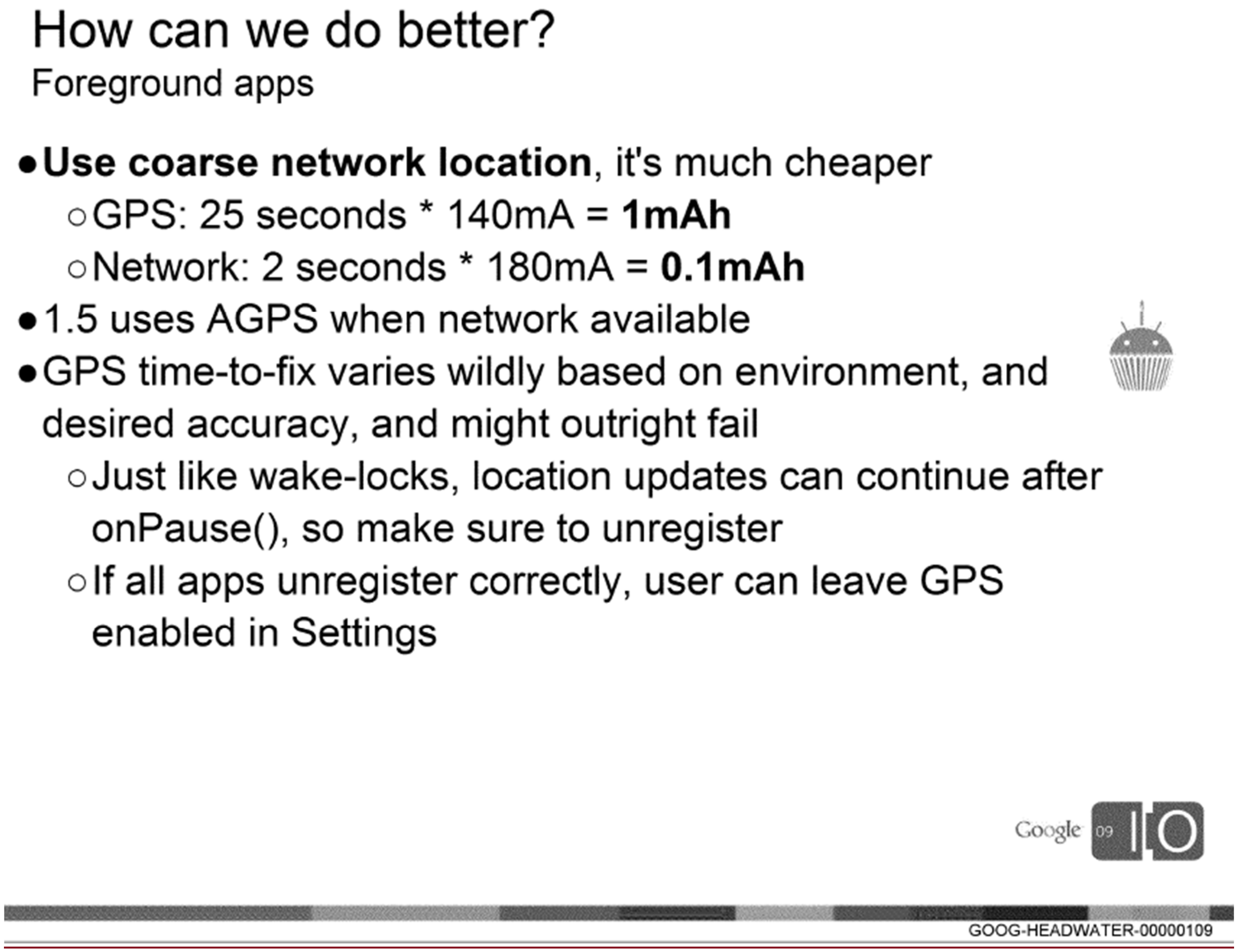
```
// Only update if WiFi or 3G is connected and not roaming
int netType = info.getType();
int netSubtype = info.getSubtype();

if (netType == ConnectivityManager.TYPE_WIFI) {
    return info.isConnected();
} else if (netType == ConnectivityManager.TYPE_MOBILE
        && netSubtype == TelephonyManager.NETWORK_TYPE_UMTS
        && !mTelephony.isNetworkRoaming()) {
    return info.isConnected();
} else {
    return false;
}
```

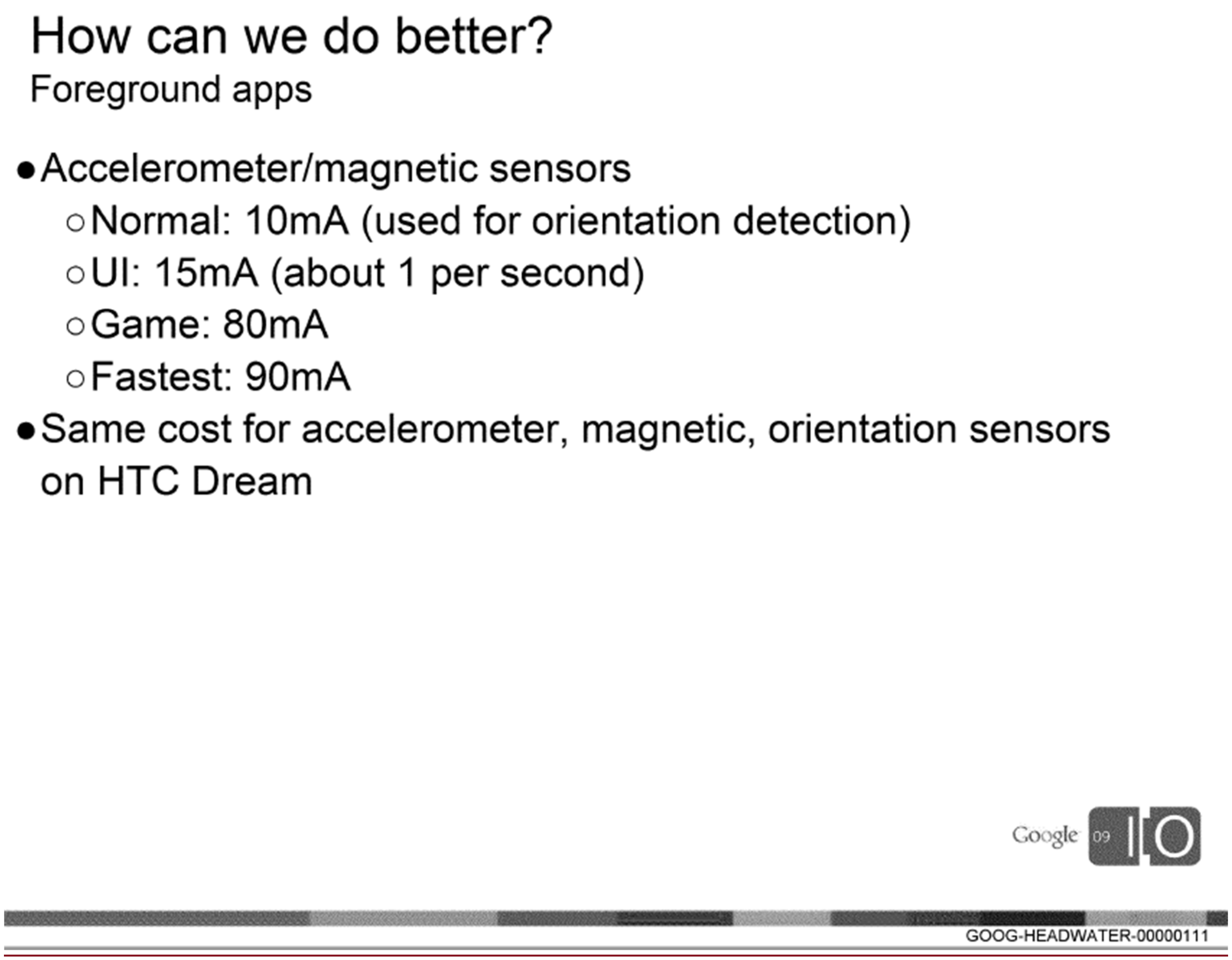
GOOG-HEADWATER-00000101

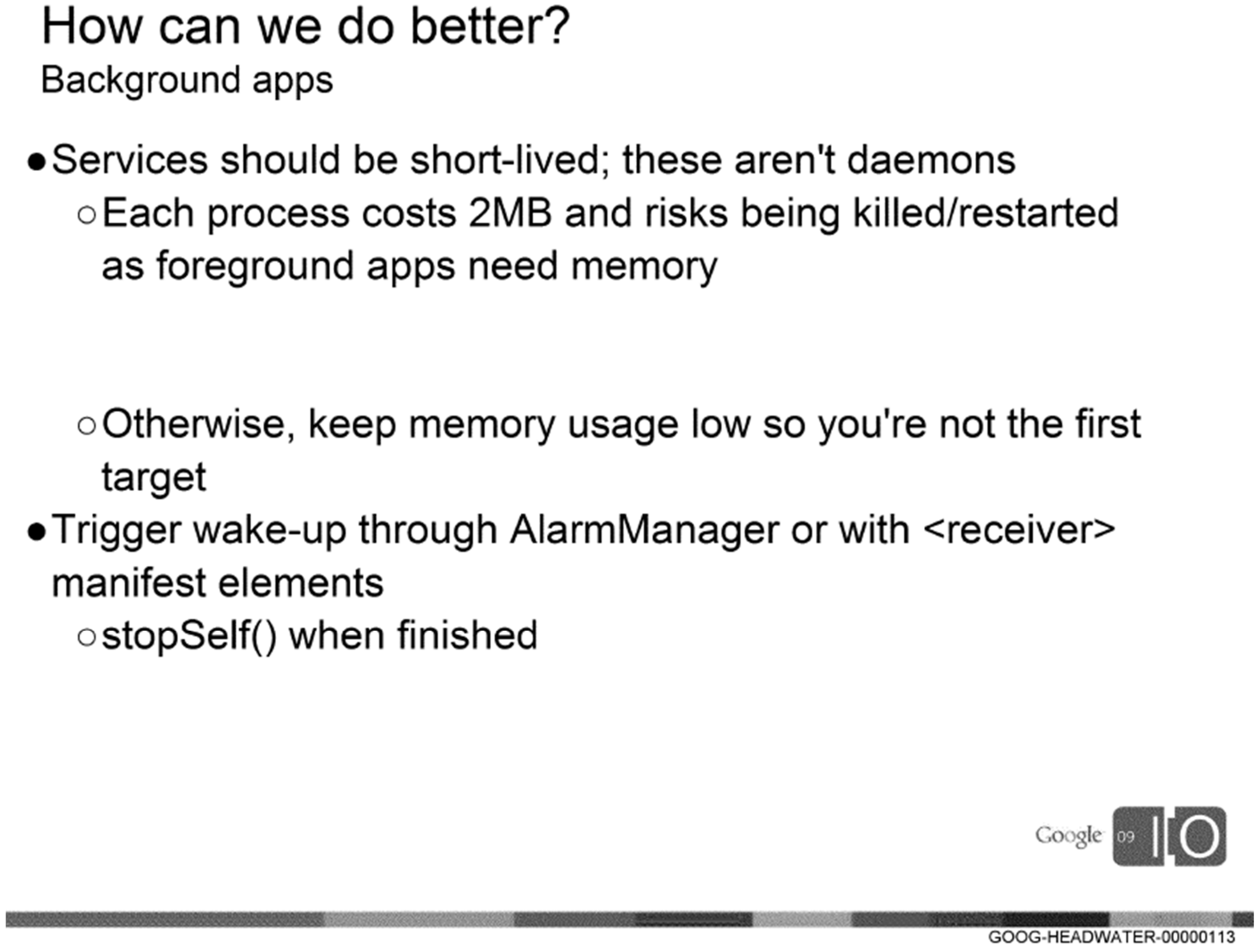
'701 Claims	Android Device with One or More Apps

	See, e.g., GOOG-HEADWATER-00000092 at 16:
	
	How can we do better?
	Foreground apps
	• **Wakelocks are costly** if forgotten
	○ Pick the lowest level possible, and use specific timeouts to work around unforseen bugs
	○ Consider using android:keepScreenOn to ensure correctness
	`<LinearLayout`
	`    android:orientation="vertical"`
	`    android:layout_width="fill_parent"`
	`    android:layout_height="fill_parent"`
	`    android:keepScreenOn="true">`
	
	GOOG-HEADWATER-00000107

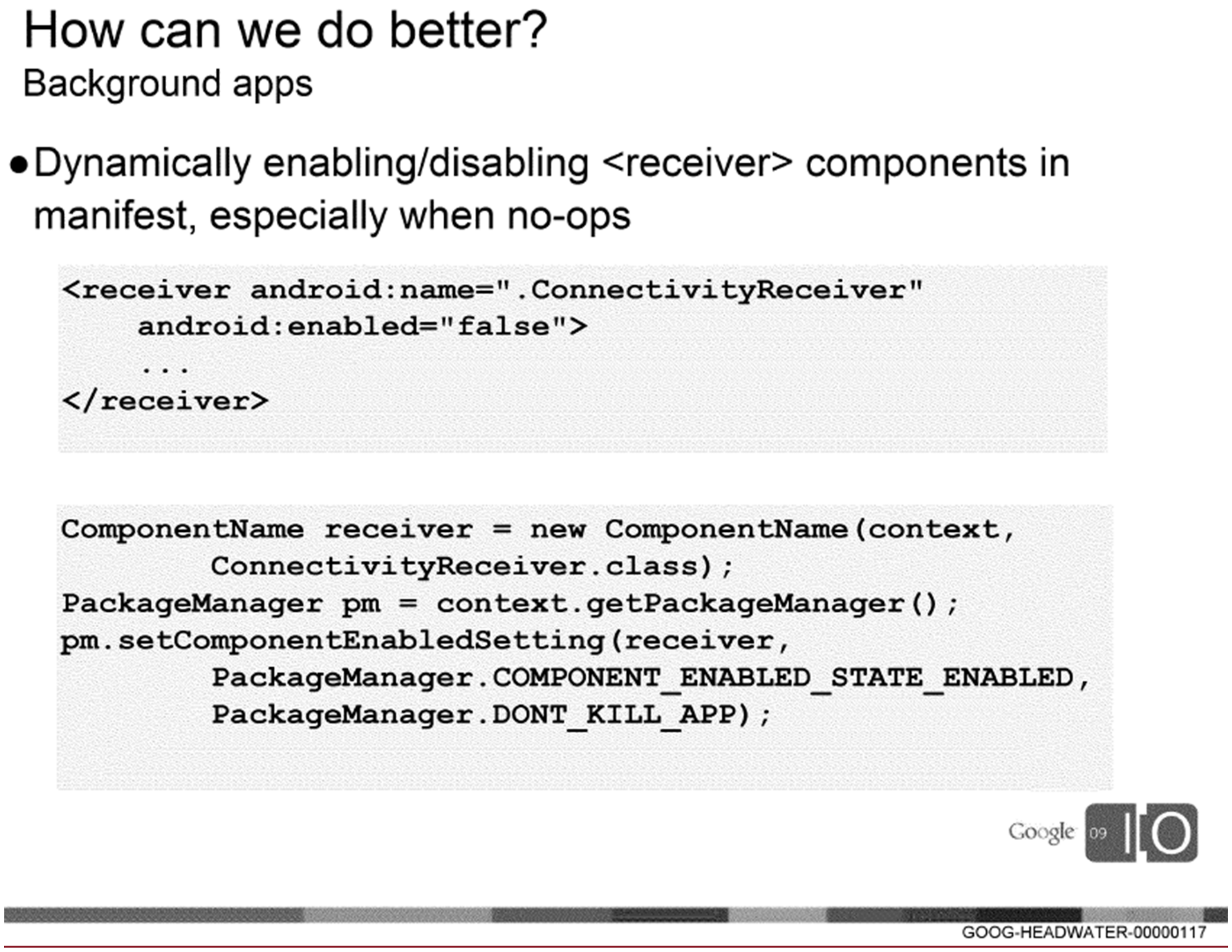
'701 Claims	Android Device with One or More Apps
	See, e.g., GOOG-HEADWATER-00000092 at 18:

’701 Claims	Android Device with One or More Apps
	 How can we do better? Foreground apps ● **Use coarse network location**, it's much cheaper ○ GPS: 25 seconds * 140mA = **1mAh** ○ Network: 2 seconds * 180mA = **0.1mAh** ● 1.5 uses AGPS when network available ● GPS time-to-fix varies wildly based on environment, and desired accuracy, and might outright fail ○ Just like wake-locks, location updates can continue after onPause(), so make sure to unregister ○ If all apps unregister correctly, user can leave GPS enabled in Settings GOOG-HEADWATER-00000109 *See, e.g.*, GOOG-HEADWATER-00000092 at 20:

'701 Claims	Android Device with One or More Apps
	 How can we do better? Foreground apps ● Accelerometer/magnetic sensors ○ Normal: 10mA (used for orientation detection) ○ UI: 15mA (about 1 per second) ○ Game: 80mA ○ Fastest: 90mA ● Same cost for accelerometer, magnetic, orientation sensors on HTC Dream Google 09 IO GOOG-HEADWATER-00000111 *See, e.g.*, GOOG-HEADWATER-00000092 at 22:

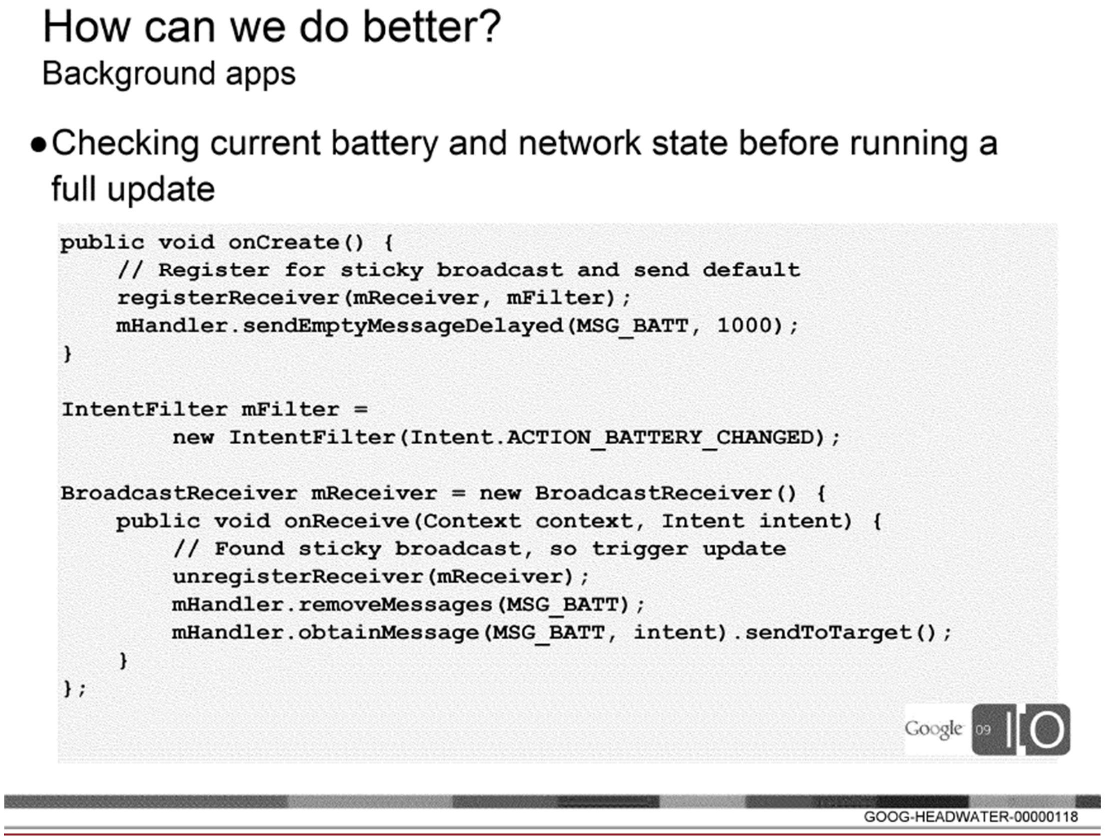
’701 Claims	Android Device with One or More Apps
	 How can we do better? Background apps ● Services should be short-lived; these aren't daemons ○ Each process costs 2MB and risks being killed/restarted as foreground apps need memory ○ Otherwise, keep memory usage low so you're not the first target ● Trigger wake-up through AlarmManager or with <receiver> manifest elements ○ stopSelf() when finished Google 09 IO GOOG-HEADWATER-00000113 *See, e.g.*, GOOG-HEADWATER-00000092 at 26:

'701 Claims	Android Device with One or More Apps

	
	How can we do better?
	Background apps
	● Dynamically enabling/disabling <receiver> components in manifest, especially when no-ops
	`<receiver android:name=".ConnectivityReceiver"`
	`    android:enabled="false">`
	`    ...`
	`</receiver>`
	`ComponentName receiver = new ComponentName(context,`
	`        ConnectivityReceiver.class);`
	`PackageManager pm = context.getPackageManager();`
	`pm.setComponentEnabledSetting(receiver,`
	`        PackageManager.COMPONENT_ENABLED_STATE_ENABLED,`
	`        PackageManager.DONT_KILL_APP);`
	Google 09 IO
	GOOG-HEADWATER-00000117

<table>
<tr><th>'701 Claims</th><th>Android Device with One or More Apps</th></tr>
<tr><td></td><td>

See, e.g., GOOG-HEADWATER-00000092 at 27:



How can we do better?

Background apps

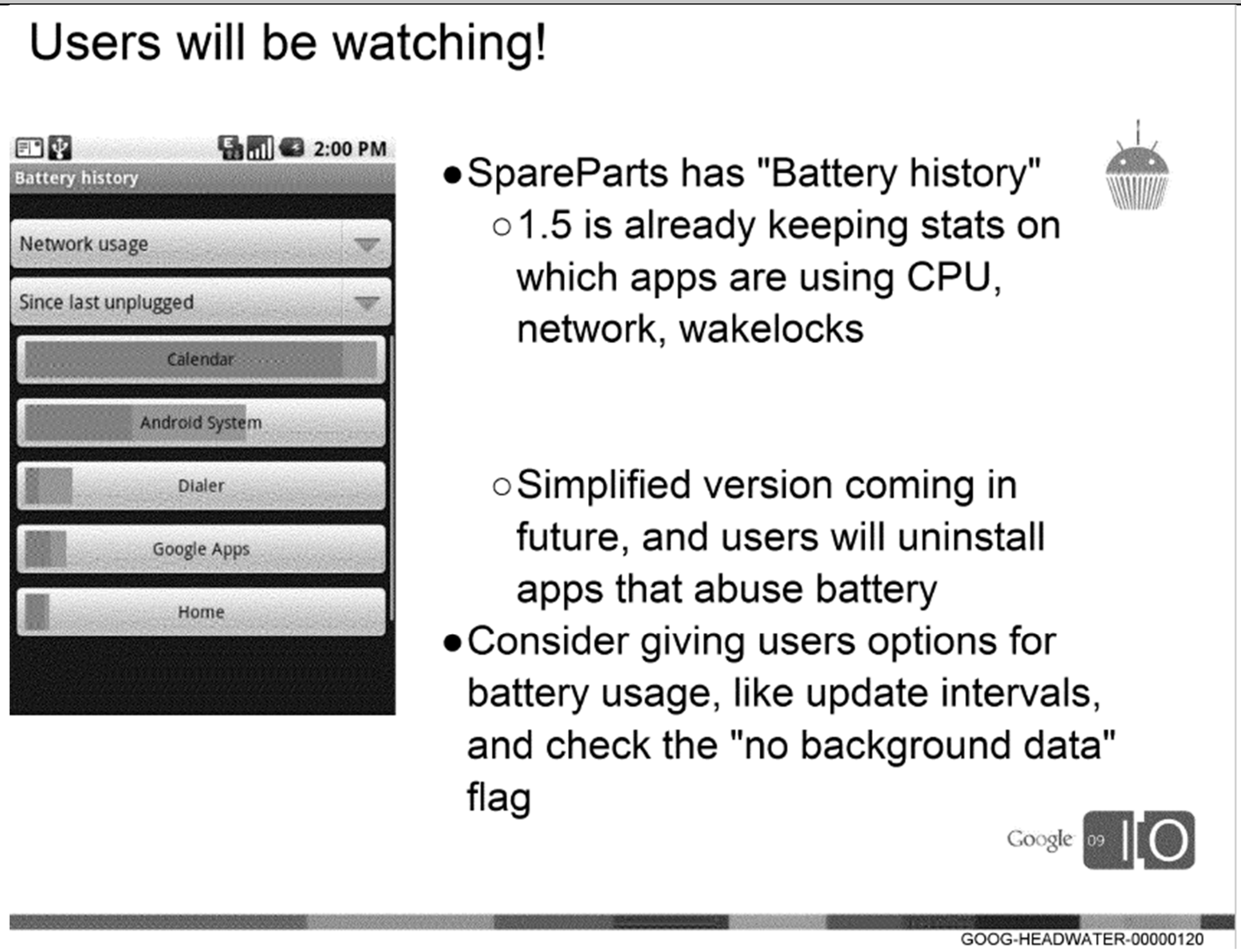
- Checking current battery and network state before running a full update

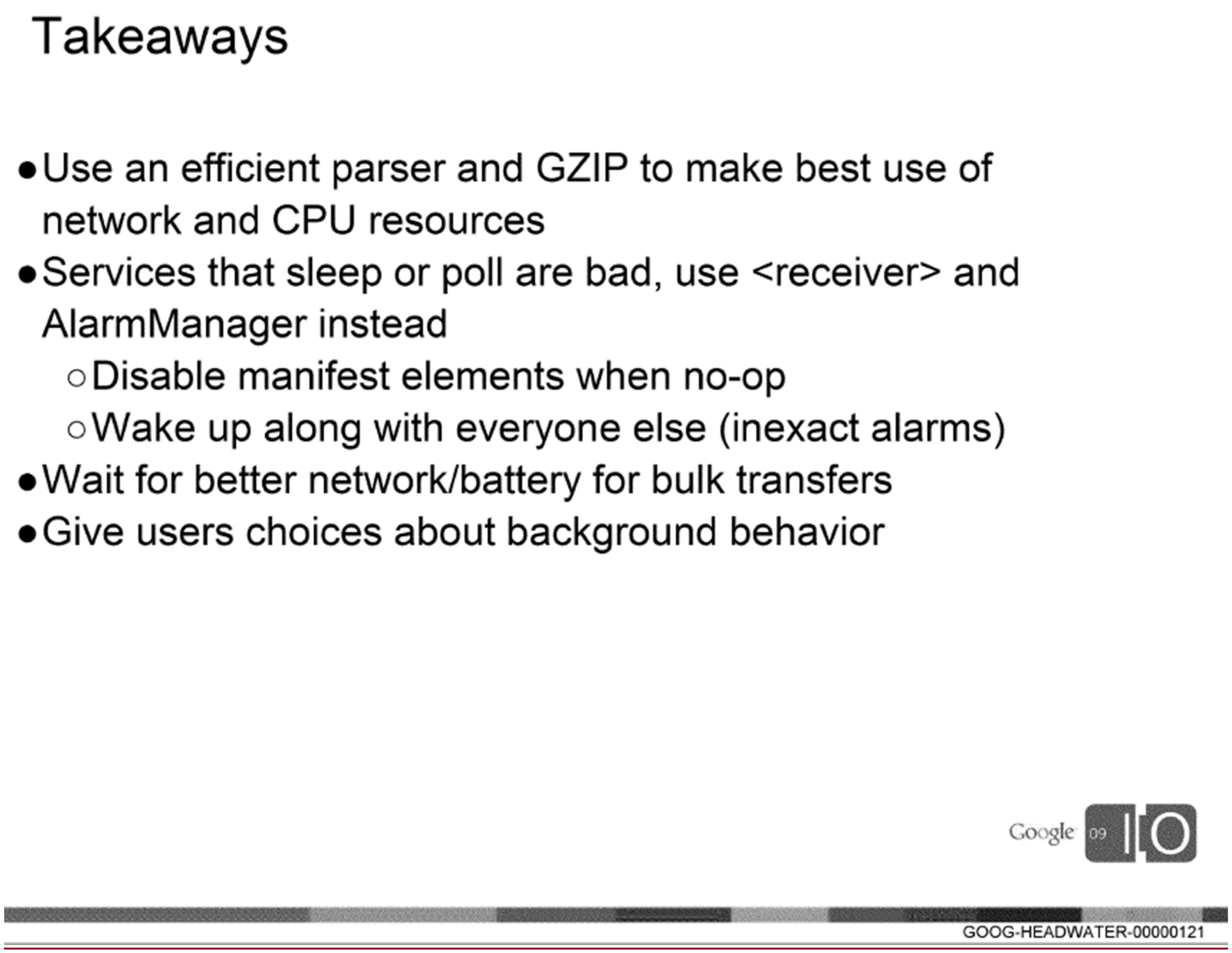
```
public void onCreate() {
    // Register for sticky broadcast and send default
    registerReceiver(mReceiver, mFilter);
    mHandler.sendEmptyMessageDelayed(MSG_BATT, 1000);
}

IntentFilter mFilter =
        new IntentFilter(Intent.ACTION_BATTERY_CHANGED);

BroadcastReceiver mReceiver = new BroadcastReceiver() {
    public void onReceive(Context context, Intent intent) {
        // Found sticky broadcast, so trigger update
        unregisterReceiver(mReceiver);
        mHandler.removeMessages(MSG_BATT);
        mHandler.obtainMessage(MSG_BATT, intent).sendToTarget();
    }
};
```

GOOG-HEADWATER-00000118

See, e.g., GOOG-HEADWATER-00000092 at 29:

</td></tr>
</table>

'701 Claims	Android Device with One or More Apps
	 Users will be watching! 2:00 PM Battery history Network usage Since last unplugged Calendar Android System Dialer Google Apps Home Google 09 IO ● SpareParts has "Battery history" ○ 1.5 is already keeping stats on which apps are using CPU, network, wakelocks ○ Simplified version coming in future, and users will uninstall apps that abuse battery ● Consider giving users options for battery usage, like update intervals, and check the "no background data" flag GOOG-HEADWATER-00000120 *See, e.g.*, GOOG-HEADWATER-00000092 at 30:

'701 Claims	Android Device with One or More Apps
	 Takeaways ● Use an efficient parser and GZIP to make best use of network and CPU resources ● Services that sleep or poll are bad, use <receiver> and AlarmManager instead ○ Disable manifest elements when no-op ○ Wake up along with everyone else (inexact alarms) ● Wait for better network/battery for bulk transfers ● Give users choices about background behavior GOOG-HEADWATER-00000121
[13] The wireless end-user device of claim 1,	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures:

'701 Claims	**Android Device with One or More Apps**
wherein the one or more processors are further configured to dynamically change the application of the first differential traffic control policy based on a device usage state.	*See*, *e.g.*, the disclosures identified for claims [1 pre] – [1e], [12].
[14] The wireless end-user device of claim 1, wherein the one or more processors are further configured to dynamically change the application of the first differential traffic control policy based on monitoring of user interaction with the device.	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See*, *e.g.*, the disclosures identified for claims [1 pre] – [1e].
[15] The wireless end-user device of claim 1, wherein the one or more processors are further configured to dynamically change the application of the first differential traffic	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See*, *e.g.*, the disclosures identified for claims [1 pre] – [1e], [12].

’701 Claims	Android Device with One or More Apps
control policy based on power control state changes for one or more of the modems.	
[16] The wireless end-user device of claim 1, wherein the one or more processors are configured to associate the first end-user application with the differential traffic control policy based on an application behavior.	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for claims [1 pre] – [1e].
[17] The wireless end-user device of claim 1, wherein the differential traffic control policy defines that applications to which the policy applies can only have WWAN network access events during particular time windows.	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for claims [1 pre] – [1e].
[18] The wireless end-user device of claim 1, wherein the one or more	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures:

’701 Claims	Android Device with One or More Apps
processors are further configured to update the first differential traffic control policy based on information received from a network element.	*See*, *e.g.*, the disclosures identified for claims [1 pre] – [1e].
[19] The wireless end-user device of claim 1, further comprising an agent to block, modify, remove, or replace user interface messages generated by the first end-user application based on the applied differential traffic control policy.	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See*, *e.g.*, the disclosures identified for claims [1 pre] – [1e].
[20] The wireless end-user device of claim 1, wherein the one or more processors configured to control application access are configured to selectively block access by the first end-user application by intercepting open, connect, and/or write	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See*, *e.g.*, the disclosures identified for claims [1 pre] – [1e], [9].~~.~~

'701 Claims	Android Device with One or More Apps
requests by the first end-user application to a network stack.	
[21] The wireless end-user device of claim 20, wherein the API responds to an intercepted request by the first end-user application by emulating network messaging.	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for claims [1 pre] – [1e], [9].
[22] The wireless end-user device of claim 21, wherein emulating network messaging comprises responding to a network request from the particular application by blocking the request from passing to a network stack and returning to the particular application a message indicating the network request was not successful.	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for claims [1 pre] – [1e], [9].

'701 Claims	**Android Device with One or More Apps**
[23] The wireless end-user device of claim 1, the first differential traffic control policy comprising first and second sub-policies applicable respectively to Internet data service provided using the WWAN modem to connect to a home WWAN and a roaming WWAN, wherein the one or more processors are further configured to apply the first sub-policy when Internet data service is provided through a home WWAN and to apply the second sub-policy when Internet data service is provided through a roaming WWAN.	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for claims [1 pre] – [1e].
[24] The wireless end-user device of claim 1, the first differential traffic control policy comprising first,	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for claims [1 pre] – [1e].

'701 Claims	**Android Device with One or More Apps**
second, and third sub-policies applicable respectively to Internet data service provided using the WWAN modem and three different network types from the network types consisting of 2G, 3G, 4G, home, and roaming.	
[25] The wireless end-user device of claim 1, wherein the one or more processors are configured to determine when the first end-user application is running in a background state or as a foreground application based on a state of user interface priority for the application.	Android Device with One or More Apps discloses and/or renders obvious this limitation. For example, see the following passages and/or figures, as well as related disclosures: *See, e.g.*, the disclosures identified for claims [1 pre] – [1e].